**BioRxiv**
**12 December 2019**

# RamaNet: Computational *de novo* helical protein backbone design using a long short-term memory generative adversarial neural network

Sari Sabban[1]*, Mikhail Markovsky[2]

**Abstract**

The ability to perform *de novo* protein design will allow researchers to expand the pool and variety of available proteins, by designing synthetic structures computationally they can utilise more structures than is available in the Protein Data Bank, design structures that are not found in nature, or direct the design of proteins to acquire a specific desired structure. While some researchers attempt to design proteins from first physical and thermodynamic principals, we decided to attempt to test whether it is possible to perform *de novo* helical protein design of just the backbone statistically using machine learning by building a model that used a long short-term memory generative adversarial neural network architecture. The LSTM based GAN model used only the $\phi$ and $\psi$ angles of each residue from an augmented dataset of only helical protein structures. Though the network's generated backbone structures were not perfect, they were idealised and evaluated post generation where the non-ideal structures were filtered out and the adequate structures kept. The results were successful in developing a logical, rigid, compact, helical protein backbone topology. This paper is a proof of concept that shows it is possible to generate a novel helical backbone topology using an LSTM-GAN architecture using only the $\phi$ and $\psi$ angles as features. The next step is to attempt to use these backbone topologies and sequence design them to form complete protein structures.

**Keywords**

*De novo* protein design — Neural Networks — Generative Adversarial Neural Network — Long Short-Term Memory Neural Network

[1] *Department of Biological Sciences/Faculty of Science, King Abdulaziz University, Jeddah, Makka, Kingdom of Saudi Arabia*
[1] *Electrical and Computer Engineering Department/College of Engineering, Effat University, Jeddah, Makka, Kingdom of Saudi Arabia*
[2] *North Caucasian Federal Scientific Center of Horticulture, Viticulture, Wine-making, Krasnodar, Russian Federation*
***Corresponding author**: sari.sabban@gmail.com

## Author summary

This research project stemmed from the desire to expand the pool of available protein structures that can be used as a scaffold in computational vaccine design since the number of structures available from the Protein Data Bank was not sufficient to allow for great diversity and increase the probability of grafting a target motif onto a protein scaffold. Since a protein structure's backbone can be defined by the $\phi$ and $\psi$ angles of each amino acid in the polypeptide and can effectively translate a protein's 3D structure into a table of numbers, and since protein structures are not random, this numerical representation of protein structures can be used to train a neural network to mathematically generalise what a protein structure is, and therefore use this generalisation to generate new protein structures. Instead of using all proteins in the Protein Data Bank a curated dataset was used encompassing protein structures with specific characteristics that will, theoretically, allow them to be easily evaluated computationally and chemically. This paper details how a trained neural network was able to successfully generate logical helical protein backbone structures.

## Introduction

The concept of amino acid sequences folding into globular protein molecules allows for proteins' large functional diversity, making them mediate all the functional aspects of living organisms, thus winning themselves attention from biochemists for decades. Everything in this universe can be described by a mathematical algorithm, including living organisms and their functional units. The fusion of machine learning with computational biology is accelerating research in both fields and bringing humanity closer to the setup of performing most biological research quickly, cheaply, and safely *in silico* while only translating the very crucial aspects of it. Having access to a large database of protein crystal structures would naturally result in the use of machine learning to design proteins computationally.

*De novo* protein design (i.e from the beginning) is very

well explained in this review [1]. Proteins fold into a specific shape depending on the sequence of their amino acids, and of course shape dictates function. The driving forces that allows proteins to fold are the hydrogen bond interactions within the backbone and between the side chains, the Van der Waals forces, and principally the interaction of hydrophobic side chains within the core. The space of all possible sequences for all protein sizes is extremely large (as an example there are $20^{200}$ possibilities for a 200 residue protein). Thus is it not surprising that natural proteins exist in clusters close to each other, which is logical since proteins would evolve away from a central functional protein to fold correctly and acquire new folds and functions rather than go through the tedious ordeal of finding a totally new protein structure within the space of all possibilities. Thus, even though the protein data bank adds about 10,000 new structures to its repository every year, most of these new structures are not unique folds.

The relationship between the sequence of a protein and its specific structure is understood, but we still lack a unified absolute solution to calculate one from the other. Hence why some research groups generated man-made protein designs through evolving already natural proteins [2] since randomly finding a functionally folded protein from the space of all possible protein sequences is more or less impossible. On the other hand, other researchers attempted *de novo* protein design by designing a topology from assembling short sequence peptide fragments taken from natural protein crystal structures [4] [5], these fragments are calculated statistically depending on the secondary structures they are found in. Sometimes this fragment system is combined with first physical principals to model the loops between secondary structures to achieve a desired three dimensional topology [6]. Others have used parametric equations to study and specify the desired protein geometry [7] [8] [9] [10] [11] [12]. These solutions employ an energy function, such as REF15, that uses some fundamental physical theories, statistical mechanical models, and observations of protein structures in-order to approximate the potential energy of a protein [3]. Knowing the protein potential energy allows us to guide our search for the structure of a protein given its sequence (the structure resides at the global energy minima of that protein sequence) thus attempting to connect the sequence of a protein with its structure. The critical item is in the energy function, the higher its accuracy the higher our confidence in knowing the computed structure is the real natural structure. Thus using the energy function to perform structure prediction (going from a known sequence to find the unknown three dimensional structure) can also be used to perform fixed-backbone design (going from a known three dimensional structure to find the sequence that folds it). Where as *de novo* design (neither backbone nor sequence is known), knowing one results in finding the other using the same energy function [1], and a good starting point is to design the backbone.

Other researchers have used machine learning for protein design, employing the constraints (C$\alpha$-C$\alpha$ distances) as the input feature of the network and using a sliding window to read a sequence of residues, getting their types and constraints then predicting the next one giving the output prediction as an amino acid sequence [13], this architecture reported an accuracy of 38.3%.

The deep neural network architecture that we chose was a Long Short-Term Memory (LSTM) based Generative Adversarial Network (GAN) [14]. The LSTM is usually used in natural language and data sequence processing, but in our model the LSTM was setup in a slightly different manner. The model was made up of two networks that worked against each other, the first was a generator network that was made up of a stack of LSTM layers, followed by fully connected layers, followed by a Mixture Density Network (MDN) and worked by using random noise numbers as input to build the values for the $\phi$ and $\psi$ angles. The other network was a discriminator that was made up of a stack of LSTM layers followed by fully connected layers and worked to studying the dataset and determining whether the output from the generator was a truly logical structure or not (fake or real) [15].

Our effort in this paper was to use machine learning to learn the general fold of natural proteins, and using this generalising statistical concept we can design a novel yet logical protein backbone topologies, thus getting the three dimensional structure. Our research at this moment was a proof of concept and only concerned with getting a new and unique folded ideal helical protein backbone rather than a protein with a specific sequence, or function, or a specific structure, thus our system resulted in random yet compact helical backbone topologies.

## Materials and methods

The following steps were used to generate the augmented training dataset, along with details of the neural network architecture and how the output was optimised then evaluated.

### Data generation

The entire PDB database was downloaded on 28$^{th}$ June 2018 ($\sim$150,000 structures), each entry was divided into its constituent chains resulting in individual separate structures (i.e: each PDB file had only a single chain). Each structure was analysed and chosen only if it contained the following criteria: contained only polypeptides, had a size between 80 and 150 amino acids without any breaks in the chain (a continuous polypeptide), a sum of residues that made up helices and sheets were larger than the sum of amino acids that made up loops, and the final structure having an Rg (radius of gyration) value of less than 15 88 Å. The chosen structures were then further filtered by a human to ensure only the desired structure concepts were selected, removing the structures that slipped through the initial computational filter. Furthermore, a diversity of structure folds were acheived rather than numerous repeats of the same fold (the haemoglobin fold was quite abundant). In previous attempts a mixture of different structure classes were used, where some structures were only helices,

some were only sheets, and the remaining were a mix of the two. But that proved challenging in optimising the network, as such a dataset made up of only helical protein structures was chosen for this initial proof of concept. The final dataset had 607 ideal helical structures. These structures were then cleaned (non-amino acid atoms were removed) in preparation to push them through the Rosetta modelling software that only takes in polypeptide molecules.

## Data augmentation

These 607 structures were augmented using the Rosetta FastRelax protocol [16]. This protocol performs multiple cycles of packing and minimisation. In other words, it performs small slight random moved on the backbone and side chains. Its originally intended function was to move a structure slightly to find the conformation of the backbone and side chains that corresponds to the lowest energy state as per the REF15 energy function. Since the protocol performs random moves, a structure relaxed on two separate occasions will result in two molecules that look very similar with similar minimum energy scores, but technically have different $\phi$ and $\psi$ angle values. This is the concept we used to augment our structures, and each structure was relaxed 500 times to give a final dataset size of 303,500 structures.

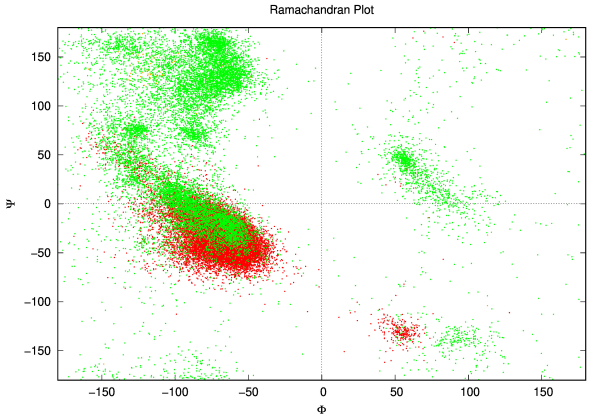


**Figure 1. Ramachandran plot of dataset.** Ramachandran plot of the dataset showing the $\phi$ and $\psi$ angles of each amino acid for each structure. This is the un-augmented data of structures that are only made of helices. Green represents the angles on amino acids in loops, while red represents the angles of amino acids in helices. Some orange can be seen where the DSSP algorithm classified the amino acids as sheets. One point to note; the angles here are represented between the range $-180°$ to $180°$ as is conventional, while in the actual dataset the range was from $0°$ to $360°$.

## Feature extraction

Using only the $\phi$ and $\psi$ angle values from a crystal structure it was possible to re-fold a structure back to its correct native fold, thus these angles were the only relevant features required to correctly fold a structure, Fig 1 details the range of angles in the un-augmented data. Each amino acid's $\phi$ and $\psi$ angle values were extracted and tabulates as in Table 1. This was the dataset used to train the neural network.

**Table 1. The PS_Helix_500.csv dataset.** The first 5 examples of the PS_Helix_500.csv dataset showing the PDB ID_chain_augmentation number, residue 1 $\phi$ angle, residue 1 $\psi$ angle, all the way to residue 150. $360.0°$ was used for the first missing angle while $0.0°$ was used to represent no residues.

| | PDB_ID | phi_1 | psi_1 | phi_2 | psi_2 | phi_3 | psi_3 | ..... | phi_150 | psi_150i |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1TQG_A_0293.pdb | 360 | 98.8 | 207.4 | 163.8 | 298.1 | 313.6 | ..... | 0.0 | 0.0 |
| 2 | 1EZ3_A_0261.pdb | 360 | 227.8 | 208.3 | 37 | 306.7 | 316.4 | ..... | 0.0 | 0.0 |
| 3 | 5IP0_E_0241.pdb | 360 | 86.7 | 293.2 | 328.7 | 292.2 | 313.1 | ..... | 0.0 | 0.0 |
| 4 | 2P5T_G_0123.pdb | 360 | 185.9 | 254.3 | 176.2 | 308.4 | 139.3 | ..... | 0.0 | 0.0 |
| 5 | 5EOH_A_0211.pdb | 360 | 144.4 | 293.5 | 334.6 | 320.9 | 320.9 | ..... | 0.0 | 0.0 |

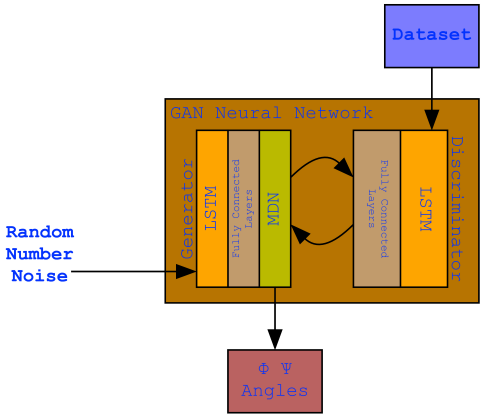


**Figure 2. The LSTM-GAN neural network.** The LSTM based GAN model employing a generative network and a discriminator network that work adversarially against each other.

## The neural network

The model in Fig 2 was built using the SenseGen model as a template [15] and consisted of two networks: a generator $G$ network and a discriminator $D$ network. The $G$ network was constructed from an LSTM layer with 64 nodes, followed by two dense fully connected MLPs with 32 nodes for the first layer and 12 nodes for the second one, both employed a sigmoid activation function:

$$sigmoid(x) = \frac{1}{1+e^{-x}}$$

Which was followed by an MDN layer employing an MDN activation function:

$$p(y|x) = \sum_{c=1}^{C} \alpha_c(x)\mathscr{D}(y|\lambda_{1,c}(x), \lambda_{2,c}(x), ...)$$

$c$: the index of the corresponding mixture component. $\alpha$: the mixing parameter. $\mathscr{D}$: the corresponding distribution to be mixed. $\lambda$: the parameters of the distribution $\mathscr{D}$, as we denote $\mathscr{D}$ to be a Gaussian distribution, $\lambda_1$ corresponds to the conditional mean and $\lambda_2$ to the conditional standard deviation. The training was done using the Adam optimiser, for each parameter $\omega^j$:

$$v_t = \rho v_{t-1} + (1-\rho) * g_t^2 \Delta\omega_t = -\frac{\eta}{\sqrt{v_t+\varepsilon}} * g_t \, \omega_{t+1} = \omega_t + \Delta\omega_t$$

$\eta$: initial learning rate. $v_t$: exponential average of squares of gradients. $g_t$: gradient at time $t$ along $\omega^j$. The Adam optimiser had an MDN activation through time loss function defined to increase the likelihood of generating the next time step value. The loss defined as the root mean squared difference between the sequence of inputs and the sequence of predictions:

$$loss = \sum_{t=1}^{T} (x_t - y_t)^2$$

$y_t$: output. $x_t$: next step sample $x_{t+1} = y_t$. The $D$ network was constructed from an LSTM layer with 64 nodes, followed by a dense fully connected MLP layer with 32 nodes, and that was followed by a single dense MLP unit layer employing a sigmoid activation function, so that the output of this network was a prediction; the probability of the data being real (indicated by the integer 1) or fake (indicated by the integer 0). The network employed the cross-entropy loss function:

$$CE = -\sum_i^C t_i log(s_i)$$

Where $t_i$ and $s_i$ are the groundtruth and the neural network score for each class $i$ in $C$. In a binary classification problem, such as the discriminator network output where $C'$= 2, the Cross Entropy Loss can be defined as:

$$CE = -\sum_{i=1}^{C'=2} t_i log(s_i) = -t_1 log(s_1) - (1-t_1) log(1-s_1)$$

Where it is assumed that there are two classes: $C_1$ and $C_2$. $t_1$ [0,1] and $s_1$ are the groundtruth and the score for $C_1$, while $t_2 = 1-t_1$ and $s_2 = 1-s_1$ are the groundtruth and the score for $C_2$. The $G$ network used random noise as a starting seed, this noise was generated by taking a single randomly distributed number between [0, 1) as the first predicted values, these values were then reshaped to the same shape of the last item of the predicted value resulting in a final shape of (batch_size, step_numbmer, 1). The network predicted the main parameters of the new value ($\mu$, $\sigma$, $\pi$) several times (according to the numbmer_of_mixtures value) and selected the single mixture randomly but according to the $\pi$ value. It then predicted the next value according to the normal distribution using the $\mu$ and $\sigma$ values. It added the final value to the prediction chain and then returned to the step 2 until the predefined sequence length was obtained. The initial random number was stripped from the returned sequence. Once the networks were constructed, the dataset was normalised and the training was done as follows for each adversarial epoch:

1. Sample minibatch from dataset (Xtrue).
2. Sample minibatch from $G$ network (XG).
3. Train the $D$ network on the training set (Xtrue, XG).
4. Sample minibatch from dataset (Xtrue).
5. Sample minibatch from $G$ network (XG).
6. Train the $G$ network on the training set (Xtrue).

The neural network had the following parameters: the $G$ learning rate was 0.001 while the $D$ learning rate was 0.0003, a drop out rate of 50% was used along with a batch size of 4 over 18,000 epochs.

### Post backbone topology generation processing and filtering

The output of the neural network was always a structure with 150 amino acids, thus the ends were trimmed if they had no helical structures. The $\phi$ and $\psi$ angel values were used to fold the structure, but since the majority of the structures ended up with helical secondary structures yet still with an open conformation, the generated structure was relaxed using the Rosetta FastRelax protocol to idealise the helices and compact the structure. Furthermore, not every prediction from the neural network resulted in an ideal structure even after the relax step, therefore we employed a filter to filter out non-ideal structures. The filter discards structures that are less than 80 amino acids, has more residues in loops than in helices, has less than 20% residues making up its core, and has a maximum distance between C$\alpha$1 and any other C$\alpha$ greater than 88 Å (the largest value in the dataset).

## Results

The dataset was named PS_Helix_500 due to the fact that the features used were the $\phi$ and $\psi$ angles, only strictly helical protein structures were used, and each structure was augmented 500 times.

The neural network was trained on the dataset for 18,000 epochs (further training collapsed the network) with a generally sloping down mean loss as shown in Fig 3 indicating that the $G$ network got better at generating data that the $D$ network classified as real rather than fake. The network was used to generate the $\phi$ and $\psi$ angles for 25 structures. All structures started with as primary straight structures with a length of 150 valine residues, the generated $\phi$ and $\psi$ angle profiles were applied to those primary structures resulting in folded structures clearly showing helical secondary structures, but the loops were usually non-ideal, thus the structures did not come together into compact topologies. To correct this, the structures were relaxed and brought together resulting in logical backbone topologies with valines as a temporary placeholder sequence. Fig 7 shows the ramachandran plot of the 25 generated structures after the relaxation step showing in red the amino acids withing helices having angles clustering around the same location as Fig 1 within the fourth quadrant as is desired for an $\alpha$-helix that has ideal angles around ($-60°$, $-45°$), our results had an angle range for the helices ($-127.4° < \phi < -44.7°$, $-71.3° < \psi < 30.6°$) not including the outliers. This setup was not perfect at achieving an ideal

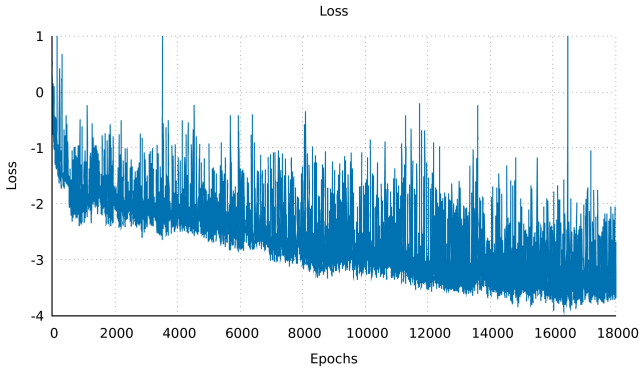


**Figure 3. The training loss.** The mean loss of the whole network over epoch, for 18,000 epochs, showing a general downward trend, this indicates that subsequent epochs the $G$ network gets better at generating structures that the $D$ network correctly classifies as real logical structures.

structure every time, so a filter was deployed to filter out suboptimal structures by choosing a structure that had more residues within helices than within loops, was not smaller than 80 residues, and had more than 20% residues comprising its core. Generating multiple structures revealed that the success rate of the network was ∼1.311% Fig 4 which took between 4 minutes and 12 hours to generate a single structure with the desired characteristics. The protocol is summarised in Fig 5, and the results are compiled in Fig 6.

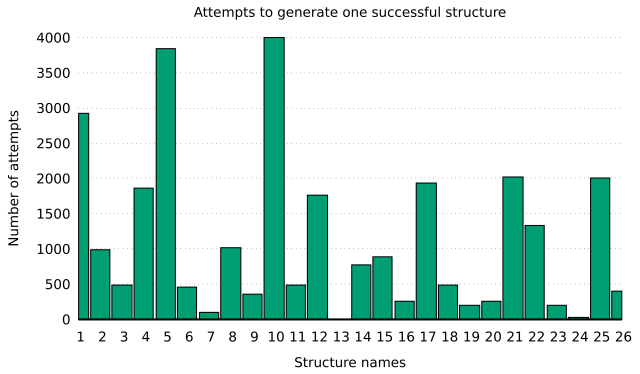


**Figure 4. Attempts to success.** The number of attempts before a structure with desired structural characteristics is reached. The figure shows great variability ranging rom 4 attempts to 3,995 attempts over the span of 26 different structures, giving an average success rate of ∼1.311%.

These 25 structures had on average 84.7% of their amino acids comprising their heliceas, along with and an average of 29.9% of thier amino acids comprising their cores.

## Conclusion

In this paper we outlined how we developed a neural network architecture that can design a helical and compact protein backbone topology. We prooved that using only the $\phi$ and $\psi$ angles of a protein structure were adequate features to design a protein backbone topology only without a sequence. Though our network had a low success rate (∼1.311%), generating multiple structures and auto filtering the suboptimal ones proved an adequate setup to achieve our goal of *de novo* helical protein backbone design within a reasonable time (1-6 hours). Still the low success rate is not satisfactory, and we are currently working on an improved model that uses further dataset dimensions to increase the success rate. As a next step, we will improve the neural network's output by implementing additional features (such as C$\alpha$ distances - sometimes referred to as constraints) to generate better loops and cores, implement sequence design, as well as crystallise the output structures to get a definitive understanding of the neural network capabilities.

## Acknowledgments

The corresponding author would like to thank the High Performance Computing Centre at King Abdulaziz University for making available the Aziz high performance computer where the corresponding author was able to augment the dataset and perform the Abinitio folding simulations.

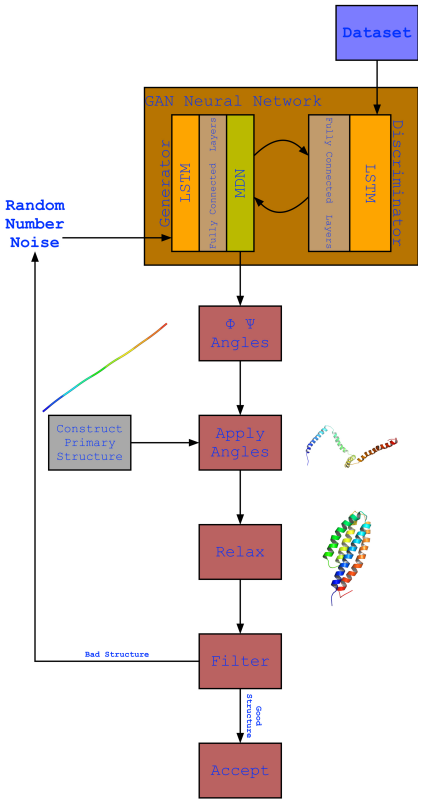


**Figure 5. The /textitde novo helical protein backbone design protocol.** The full protocol showing the structure of the model and its output. The network's output were the generated $\phi$ and $\psi$ angles which were applied to a primary structure that resulted in the development of the secondary structures but not a final compact structure due to suboptimal loop structures as a result of their variability in the dataset. To overcome this, the structure was relaxed to bring the secondary structure helices together. This did result in more compact structures but was not always ideal, thus a filter was used to filter our non-ideal structures and keep an ideal structure when generated which was kept.

## References

[1] Huang PS, Boyken SE, Baker D. The coming of age of de novo protein design. Nature. 2016;537(7620):320–7. doi:10.1038/nature19946.

[2] Dougherty MJ, Arnold FH. Directed evolution: new parts and optimized function. Curr Opin Biotechnol. 2009;20(4):486–91. doi:10.1016/j.copbio.2009.08.005.

[3] Alford RF, Leaver-Fay A, Jeliazkov JR, O'Meara MJ, DiMaio FP, Park H, et al. The Rosetta All-Atom Energy Function for Macromolecular Modeling and De-

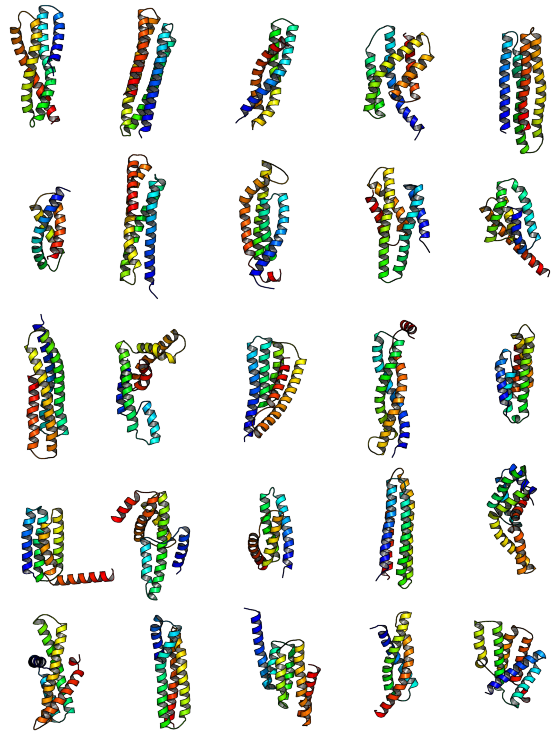

**Figure 6. Summery of five designed proteins using this method.** This figure shows all 25 structures that were generated [36]. It can be seen that all structures hare compact helical structures of variable topologies.

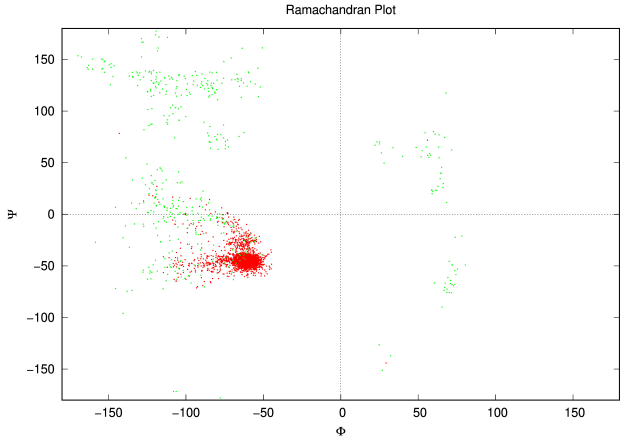


**Figure 7. Ramachandran plot of output results.** The network's output $\phi$ and $\psi$ angles for 26 structures after the relaxation step. The green dots represent the angles of amino acids within loops, and red within helices clustering around the same location as Fig 1 within the fourth quadrant as is desired for an $\alpha$-helix that has ideal angles around $(-60°, -45°)$. These structures culminated to all 25 structures in Fig 6, and had an angle range for the helices $(-127.4° < \phi < -44.7°, -71.3° < \psi < 30.6°)$ not including the outliers.

sign. J Chem Theory Comput. 2017;13(6):3031–3048. doi:10.1021/acs.jctc.7b00125.

[4] Kuhlman B, Dantas G, Ireton GC, Varani G, Stoddard BL, Baker D. Design of a novel globular protein fold with atomic-level accuracy. Science. 2003;302(5649):1364–8. doi:10.1126/science.1089427.

[5] Huang PS, Ban YE, Richter F, Andre I, Vernon R, Schief WR, et al. RosettaRemodel: a generalized framework for flexible backbone protein design. PLoS One. 2011;6(8):e24109. doi:10.1371/journal.pone.0024109.

[6] Koga N, Tatsumi-Koga R, Liu G, Xiao R, Acton TB, Montelione GT, et al. Principles for designing ideal protein structures. Nature. 2012;491(7423):222–7. doi:10.1038/nature11600.

[7] Regan L, DeGrado WF. Characterization of a helical protein designed from first principles. Science. 1988;241(4868):976–8.

[8] Harbury PB, Plecs JJ, Tidor B, Alber T, Kim PS. High-resolution protein design with backbone freedom. Science. 1998;282(5393):1462–7.

[9] Grigoryan G, Degrado WF. Probing designability via a generalized model of helical bundle geometry. J Mol Biol. 2011;405(4):1079–100. doi:10.1016/j.jmb.2010.08.058.

[10] Huang PS, Oberdorfer G, Xu C, Pei XY, Nannenga BL, Rogers JM, et al. High thermodynamic stability of parametrically designed helical bundles. Science. 2014;346(6208):481–485. doi:10.1126/science.1257481.

[11] Joh NH, Wang T, Bhate MP, Acharya R, Wu Y, Grabe M, et al. De novo design of a transmembrane Zn(2)(+)-transporting four-helix bundle. Science. 2014;346(6216):1520–4. doi:10.1126/science.1261172.

[12] Thomson AR, Wood CW, Burton AJ, Bartlett GJ, Sessions RB, Brady RL, et al. Computational design of water-soluble alpha-helical barrels. Science. 2014;346(6208):485–8. doi:10.1126/science.1257452.

[13] Wang J, Cao H, Zhang JZH, Qi Y. Computational Protein Design with Deep Learning Neural Networks. Sci Rep. 2018;8(1):6349. doi:10.1038/s41598-018-24760-x.

[14] Radford A, Metz L, Chintala S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv. 2015;.

[15] Alzantot M, Chakraborty S, Srivastava M. Sensegen: A deep learning architecture for synthetic sensor data generation. In: 2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops). IEEE; 2017. p. 188–193.

[16] Tyka MD, Keedy DA, Andre I, Dimaio F, Song Y, Richardson DC, et al. Alternate states of proteins revealed by detailed energy landscape mapping. J Mol Biol. 2011;405(2):607–18. doi:10.1016/j.jmb.2010.11.008.

[17] Kuhlman B, Dantas G, Ireton GC, Varani G, Stoddard BL, Baker D. Design of a novel globular protein fold with atomic-level accuracy. Science. 2003;302(5649):1364–8. doi:10.1126/science.1089427.

[18] Murphy GS, Mills JL, Miley MJ, Machius M, Szyperski T, Kuhlman B. Increasing sequence diversity with flexible

backbone protein design: the complete redesign of a protein hydrophobic core. Structure. 2012;20(6):1086–96. doi:10.1016/j.str.2012.03.026.

[19] Chaudhury S, Lyskov S, Gray JJ. PyRosetta: a script-based interface for implementing molecular modeling algorithms using Rosetta. Bioinformatics. 2010;26(5):689–91. doi:10.1093/bioinformatics/btq007.

[20] Cock PJ, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, et al. Biopython: freely available Python tools for computational molecular biology and bioinformatics. Bioinformatics. 2009;25(11):1422–3. doi:10.1093/bioinformatics/btp163.

[21] Joosten RP, te Beek TA, Krieger E, Hekkelman ML, Hooft RW, Schneider R, et al. A series of PDB related databases for everyday needs. Nucleic Acids Res. 2011;39(Database issue):D411–9. doi:10.1093/nar/gkq1105.

[22] Baakman C, Krieger E, Black J, Touw WG, Vriend G, Joosten RP, et al. A series of PDB-related databanks for everyday needs. Nucleic Acids Research. 2014;43(D1):D364–D368. doi:10.1093/nar/gku1028.

[23] Buchan DW, Minneci F, Nugent TC, Bryson K, Jones DT. Scalable web services for the PSIPRED Protein Analysis Workbench. Nucleic Acids Res. 2013;41(Web Server issue):W349–57. doi:10.1093/nar/gkt381.

[24] Slabinski L, Jaroszewski L, Rychlewski L, Wilson IA, Lesley SA, Godzik A. XtalPred: a web server for prediction of protein crystallizability. Bioinformatics. 2007;23(24):3403–5. doi:10.1093/bioinformatics/btm477.

[25] Rohl CA, Strauss CE, Misura KM, Baker D. Protein structure prediction using Rosetta. Methods Enzymol. 2004;383:66–93. doi:10.1016/s0076-6879(04)83004-0.

[26] Gront D, Kulp DW, Vernon RM, Strauss CE, Baker D. Generalized fragment picking in Rosetta: design, protocols and applications. PLoS One. 2011;6(8):e23294. doi:10.1371/journal.pone.0023294.

[27] Song Y, DiMaio F, Wang RY, Kim D, Miles C, Brunette T, et al. High-resolution comparative modeling with RosettaCM. Structure. 2013;21(10):1735–42. doi:10.1016/j.str.2013.08.005.

[28] Raman S, Vernon R, Thompson J, Tyka M, Sadreyev R, Pei J, et al. Structure prediction for CASP8 with all-atom refinement using Rosetta. Proteins. 2009;77 Suppl 9:89–99. doi:10.1002/prot.22540.

[29] Bertoni M, Kiefer F, Biasini M, Bordoli L, Schwede T. Modeling protein quaternary structure of homo- and hetero-oligomers beyond binary interactions by homology. Sci Rep. 2017;7(1):10480. doi:10.1038/s41598-017-09654-8.

[30] Benkert P, Biasini M, Schwede T. Toward the estimation of the absolute quality of individual protein structure models. Bioinformatics. 2011;27(3):343–50. doi:10.1093/bioinformatics/btq662.

[31] Guex N, Peitsch MC, Schwede T. Automated comparative protein structure modeling with SWISS-MODEL and Swiss-PdbViewer: a historical perspective. Electrophoresis. 2009;30 Suppl 1:S162–73. doi:10.1002/elps.200900140.

[32] Bienert S, Waterhouse A, de Beer TA, Tauriello G, Studer G, Bordoli L, et al. The SWISS-MODEL Repository-new features and functionality. Nucleic Acids Res. 2017;45(D1):D313–d319. doi:10.1093/nar/gkw1132.

[33] Waterhouse A, Bertoni M, Bienert S, Studer G, Tauriello G, Gumienny R, et al. SWISS-MODEL: homology modelling of protein structures and complexes. Nucleic Acids Res. 2018;46(W1):W296–w303. doi:10.1093/nar/gky427.

[34] Kallberg M, Wang H, Wang S, Peng J, Wang Z, Lu H, et al. Template-based protein structure modeling using the RaptorX web server. Nat Protoc. 2012;7(8):1511–22. doi:10.1038/nprot.2012.085.

[35] Kelley LA, Mezulis S, Yates CM, Wass MN, Sternberg MJ. The Phyre2 web portal for protein modeling, prediction and analysis. Nat Protoc. 2015;10(6):845–58. doi:10.1038/nprot.2015.053.

[36] Schrödinger, LLC. The PyMOL Molecular Graphics System, Version 1.8; 2015.