Global-Context Aware Generative Protein Design for Structure-to-sequence Translation

Anonymous ACL submission

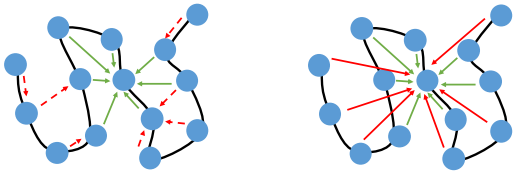
Abstract

The linear sequence of amino acids determines protein structure and function. Protein design, known as the inverse of protein structure prediction, aims to obtain a novel protein sequence that will fold into the defined structure. Recent works on computational protein design have studied designing sequences for the desired backbone structure with local positional information and achieved competitive performance. However, similar local environments in different backbone structures may result in different amino acids, which indicates the global context of protein structure matters. Thus, we propose the **G**lobal-**C**ontext **A**ware generative protein design method (GCA), consisting of *local* and *global* modules. While *local* modules focus on relationships between neighbor amino acids, *global* modules explicitly capture non-local contexts. Experimental results demonstrate that GCA achieves state-of-the-art performance on structure-based protein design. Our code and pretrained model will be released.

1 Introduction

Computational protein design, which aims to invent protein molecules with desired structures and functions automatically, has a wide range of applications in therapeutics and pharmacology (Brunette et al., 2015; Huang et al., 2016; Langan et al., 2019). Recent years have witnessed remarkable advancements (Gao et al., 2020; Kryshtafovych et al., 2019; Yang et al., 2019). While classical protein design approaches depend on composite energy functions of protein physics and sampling algorithms for exploring sequence and structure spaces, data-driven approaches apply neural networks to generate protein sequences with less prior knowledge.

Designing a protein sequence for a given structure remains challenging, as mapping the structure space to the sequence space is difficult. Current data-driven methods (Ingraham et al., 2019; Jing et al., 2021; Cao et al., 2021) agree on the assumption based on biology and physics prior knowledge that, for each amino acid, its neighborhoods have the most immediate and vital effects on itself. The majority of such methods represent protein structures as graphs with hand-crafted features and aggregate local messages. The computational protein design is formulated to learn features from 3D structures with the local message passing mechanism. However, *the similar local environment in different proteins may correspond to different amino acids*. Local neighbors do matter (Ribeiro et al., 2020), but it is not enough to obtain high-quality protein sequences.



(a) Local module (b) Global module

Figure 1: The comparison between the local module and global module. Information flows from adjacent nodes, and distant nodes are denoted as green arrows and red arrows, respectively. The red dashed arrows indicate the implicit information flow from distant nodes.

To fully explore the non-local information, we propose the **G**lobal-**C**ontext **A**ware generative protein design method (GCA) with local and global modules. While local modules are built upon graph attention networks that aggregate local messages gained from neighbors, global modules extend local graph attention to global self-attention in the form of Transformer (Vaswani et al., 2017) architectures. As shown in Fig. 1, the local module focuses on adjacent structure information though distant nodes can deliver information implicitly; the global module explicitly gathers information from distant nodes. By composing multiple blocks of local and global modules, GCA can capture high-order dependencies between protein sequences and structures in both neighbor-level and overall-level.

2 Proposed method

2.1 Preliminaries

Protein primary structure is the linear sequence of amino acids, typically notated as a string of letters. A protein sequence $\mathcal{S}^N = \{(a)^N | a \in \{A, R, N, ..V\}\}$ has N amino acids while each of them is represented by a letter of twenty possible letters such as A, R, N, D, C, Q, E, G, H, I, L, K, M, F, P, S, T, W, Y, and V. A protein sequence will fold into a protein tertiary structure $\mathcal{X}^N = \{\mathbf{x}_i^\omega \in \mathbb{R}^3 : 1 \leq i \leq N, \omega \in \{\mathrm{C}\alpha, \mathrm{C}, \mathrm{N}, \mathrm{O}\}\}$, where N is the number of amino acids and ω indicates the chain in protein. As shown in Fig. 2, protein design predicts the protein sequence of a given protein structure, while structure prediction is the opposite.

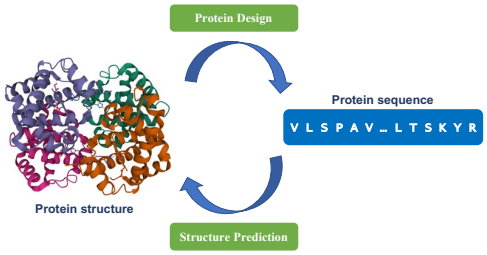


Figure 2: Two important tasks in protein modeling: structure prediction and protein design. The visualization is created by Mol* Viewer (Sehnal et al., 2021).

2.2 Represent protein as a graph

The structure of a protein is represented as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where node feature $\mathbf{v}_i \in \mathcal{V}$ corresponds to an amino acid while edge feature $\mathcal{E} = \{\mathbf{e}_{ij}\}_{j \in \mathcal{N}_i}$ suggests the rotation-invariant and translation-invariant relationships between each pair of nodes $\mathbf{v}_i$ and $\mathbf{v}_j$. In particular, $\mathcal{N}_i$ denotes the K-nearest neighbors of node i calculated by Euclidean distances of the backbone.

For node features, we construct three dihedral angles $\{\phi_i, \psi_i, \omega_i\}$ of the protein backbone from $\mathrm{C}_{i-1}, \mathrm{N}_i, \mathrm{C}\alpha_i, \mathrm{C}_i$, and N_{i+1}. Then these dihedral angels are embedded on the 3-torus as $\mathbf{v}_i = \{\sin, \cos\} \times \{\phi_i, \psi_i, \omega_i\}$.

For edge features, we focus on describing relative spatial relationships between amino acids that satisfy rotation-invariant and translation-invariant properties. To simplify the computation, we only consider the position $\mathbf{x}_i^{\mathrm{C}\alpha}$ of the alpha carbon $\mathrm{C}\alpha$ as it's the central carbon atom in each amino acid. The distance $\|\mathbf{x}_j^{C\alpha} - \mathbf{x}_i^{C\alpha}\|_2, \forall i \neq j$ is encoded by Gaussian radial basis functions $\mathbf{r}(\cdot)$. Then, as shown in Fig. 3, the direction is encoded by

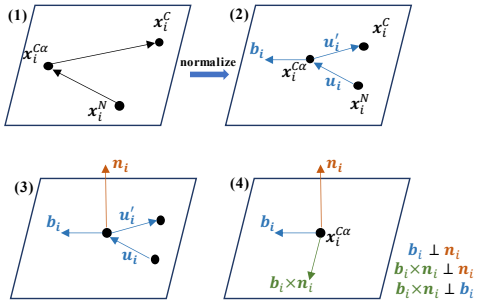


Figure 3: A general view of how the local coordinate system is built. The final coordinate satisfies $\mathbf{b}_i \perp \mathbf{n}_i, \mathbf{b}_i \times \mathbf{n}_i \perp \mathbf{n}_i, \mathbf{b}_i \times \mathbf{n}_i \perp \mathbf{b}_i$.

$\mathbf{O}_i^T \frac{\mathbf{x}_j^{C\alpha} - \mathbf{x}_i^{C\alpha}}{\|\mathbf{x}_j^{C\alpha} - \mathbf{x}_i^{C\alpha}\|}$ while $\mathbf{O}_i = [\mathbf{b}_i \quad \mathbf{n}_i \quad \mathbf{b}_i \times \mathbf{n}_i]$ defines a local coordinate system for each amino acid by:

$$\mathbf{u}_i = \frac{\mathbf{x}_i^{C\alpha} - \mathbf{x}_{i-1}^{C\alpha}}{\|\mathbf{x}_i^{C\alpha} - \mathbf{x}_{i-1}^{C\alpha}\|}, \mathbf{b}_i = \frac{\mathbf{u}_i - \mathbf{u}_{i+1}}{\|\mathbf{u}_i - \mathbf{u}_{i+1}\|}, \mathbf{n}_i = \frac{\mathbf{u}_i \times \mathbf{u}_{i+1}}{\|\mathbf{u}_i \times \mathbf{u}_{i+1}\|}. \tag{1}$$

The orientation is encoded by the common-used quaternion representation of rotation matrix $\mathbf{q}(\mathbf{O}_i^T \mathbf{O}_j)$. Thus, the edge feature $\mathbf{e}_{ij}$ is the concatenation of the distance, direction and orientation encodings as:

$$\mathbf{e}_{ij} = \left(\mathbf{r}(\|\mathbf{x}_j^{C\alpha} - \mathbf{x}_i^{C\alpha}\|_2), \mathbf{O}_i^T \frac{\mathbf{x}_j^{C\alpha} - \mathbf{x}_i^{C\alpha}}{\|\mathbf{x}_j^{C\alpha} - \mathbf{x}_i^{C\alpha}\|}, \mathbf{q}(\mathbf{O}_i^T \mathbf{O}_j)\right). \tag{2}$$

2.3 Network architecture

2.3.1 Local module

The local module is a graph neural network (GNN) that aggregates both node embeddings and local edge embeddings and updates the node embedding for further sequence generations. Considering a L-layer GNN, the key operations *aggregating* and *updating* can be formulated as follows:

$$\mathbf{h}_{\mathcal{N}_i}^{(l)} = aggregating^{(l)}(\{(\mathbf{h}_i^{(l-1)}, \mathbf{h}_j^{(l-1)}, \mathbf{h}_{\mathbf{e}_{ij}}) : j \in \mathcal{N}_i\}), \tag{3}$$

$$\mathbf{h}_i^{(l)} = updating(\mathbf{h}_i^{(l-1)}, \mathbf{h}_{\mathcal{N}_i}^{(l)}), \tag{4}$$

where $\mathbf{h}_i^{(l)} \in \mathbb{R}^D$ denotes the embedding of node i on the l-th layer, $\mathbf{h}_{\mathcal{N}_i}^{(l)} \in \mathbb{R}^{K \times D}$ denotes the local edge embedding of node i's neighbors on the l-th layer, K is the number of local neighbors, and D is the dimensions of the embedding. In particular, $\mathbf{h}_i^{(0)} \in \mathbb{R}^D$ is the embedding of $\mathbf{v}_i$, and $\mathbf{h}_{\mathbf{e}_{ij}} \in \mathbb{R}^D$ is the embedding of the edge feature $\mathbf{e}_{ij}$. The local edge information flows into node embeddings at each layer, while distant edge information flows through high-level layers.

In order to capture the relationships in local neighborhoods, we generalize graph attention scheme that take advantage of attention coefficients $\alpha \in \mathbb{R}^{N\times K}$ as strong relational inductive bias. Specifically, the attention coefficients are calculated as follows:

$$\alpha_{ij} = \frac{\exp(c_{ij})}{\sum_{k\in\mathcal{N}(i)}\exp(c_{ik})}, \forall j \in \mathcal{N}_i, \tag{5}$$

where c_{ij} is expressed as:

$$c_{ij} = \sigma\Big(a^T\big[\mathbf{W}\mathbf{h}_i^{(l-1)} \,||\, \mathbf{W}\mathbf{h}_j^{(l-1)} \,||\, \mathbf{W}\mathbf{h}_{\mathbf{e}_{ij}}\big]\Big), \forall j \in \mathcal{N}_i, \tag{6}$$

and $\mathbf{W} \in \mathbb{R}^{D\times D}$, $a \in \mathbb{R}^{3D}$ are learnable parameters, σ is the activation function, $||$ is the concatenation operation.

Thus, the *aggregating* operation is adopted as:

$$\mathbf{h}_{\mathcal{N}_i}^{(l)} = \sum_{j\in\mathcal{N}_i} \alpha_{ij}\mathbf{W}_r\big[\mathbf{h}_i^{(l-1)} \,||\, \mathbf{h}_j^{(l-1)} \,||\, \mathbf{h}_{\mathbf{e}_{ij}}\big], \tag{7}$$

where $\mathbf{W}_r \in \mathbb{R}^{D\times 3D}$ encodes the relation between i and j. The *updating* operation is simply renovating hidden layers by their local neighbors: $\mathbf{h}_i^{(l)} = \mathbf{h}_{\mathcal{N}_i}^{(l)}$.

2.3.2 Global module

The global module is the fully self-attention network that generalizes Transformer (Vaswani et al., 2017) to protein graph. Specifically, the attention coefficients are calculated as follows:

$$\alpha_{ij} = \frac{\exp(c_{ij})}{\sum_{k\in\mathcal{V}}\exp(c_{ik})}, \tag{8}$$

where c_{ij} is expressed as:

$$c_{ij} = \frac{1}{\sqrt{d}}\Big(\mathbf{W}_q\mathbf{h}_i^{(l-1)}\Big)^T\Big(\mathbf{W}_k\big[\mathbf{h}_i^{(l-1)} \,||\, \mathbf{h}_j^{(l-1)} \,||\, \mathbf{h}_{\mathbf{e}_{ij}}\big]\Big), \tag{9}$$

where $\mathbf{W}_q \in \mathbb{R}^{D\times D}$, $\mathbf{W}_k \in \mathbb{R}^{D\times 3D}$ are parameter matrices for the query and key, and d is a scale factor. Then, the *aggregating* operation is formulated as:

$$\mathbf{h}_{\mathcal{N}_i}^{(l)} = \sum_{j\in\mathcal{V}} \alpha_{ij}\mathbf{W}_r\mathbf{h}_j^{(l-1)}, \tag{10}$$

the *updating* operation is defined by employing layer normalization (LayerNorm), dropout (DropOut) and fully connected networks (FFN):

$$\mathbf{h}_i^{(l)} = \mathrm{LayerNorm}(\mathbf{h}_{\mathcal{N}_i}^{(l)} + \mathrm{DropOut}(\mathrm{FFN}(\mathbf{h}_{\mathcal{N}_i}^{(l)}))). \tag{11}$$

The overall architecture with stacked local modules and global modules is shown in Fig. 4.

3 Experiments

3.1 Experimental settings

3.1.1 Dataset

We use the CATH 4.2 dataset collected by (Ingraham et al., 2019) for evaluation. This dataset obtains full chains up to length 500, and structures have been partitioned with 40% non-redundancy by their CATH (**C**lass, **A**rchitecture, **T**opology, **H**omologous). With no CAT overlap between sets, there are 18024 chains in the training set, 608 chains in the validation set, and 1120 chains in the test set, respectively. Two subsets of the entire test set are evaluated simultaneously: a 'Short' subset containing chains up to length 100 and a 'Single chain' subset for comparing with baselines that only use the single chain. We also consider a smaller dataset TS50, which is the standard benchmark introduced by (Li et al., 2014). Though the model is still trained on the CATH 4.2 dataset, we filter the training and validation sets to ensure there is no overlap with TS50.

3.1.2 Measurement

Perplexity Following (Ingraham et al., 2019; Madani et al., 2020), we define the perplexity that evaluates the predicted protein sequences from natural language perspective:

$$\mathrm{PERP}(\mathcal{S}^N, \mathcal{X}^N) = \exp\Big(-\frac{1}{N}\sum_{i=1}^{N}\mathcal{S}_i^N \log p(\mathcal{S}_i^N \mid \mathcal{X}_i^N)\Big), \tag{12}$$

where $(\mathcal{S}^N, \mathcal{X}^N)$ is the sequence-structure pair of a protein with N amino acids. $\mathcal{S}_i^N, \mathcal{X}_i^N$ denote the i-th amino acid in sequence and structure respectively. $p(\mathcal{S}_i^N|\mathcal{X}_i^N)$ is the output probability from the model.

Recovery To evaluate the predicting accuracy of the protein sequence at per-residue level, we consider the recovery:

$$\mathrm{REC}(\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{(\mathcal{X}^N,\mathcal{S}^N)\in\mathcal{D}}\frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[S_i^N = \arg\max p(S_i^N \mid \mathcal{X}_i^N)], \tag{13}$$

where $\mathcal{D}$ denotes the whole dataset.

3.1.3 Model architecture and optimization

In all experiments, GCA model is built by three blocks of local and global modules for the encoder and decoder with the hidden dimension of 128. The Adam optimizer with learning rate of $1e^{-3}$ is employed. Models are trained for 100 epochs while each batch contains up to 2,500 characters.

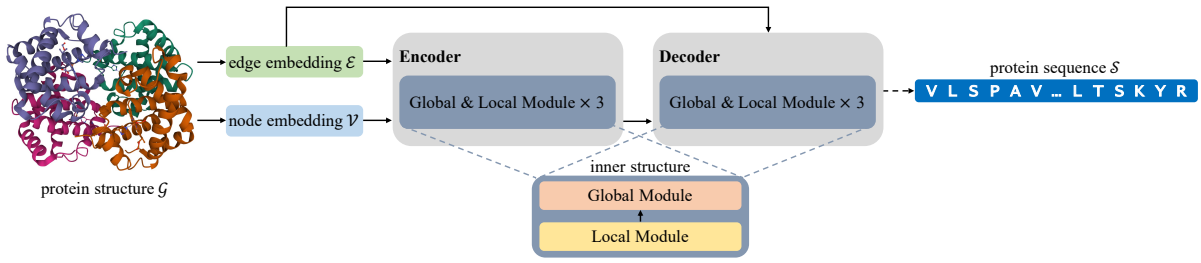


Figure 4: The architecture of our proposed method.

3.2 Experimental results

We present the median of PERP in Table 1. While the structure-free language model LSTMs produce confusing protein sequences, structure-based models obtain less-perplex protein sequences, indicating the importance of structural features. GCA outperforms other models as global contexts of protein structures are taken into account.

Methods	Short	Single chain	All
Language models			
LSTM ($h = 128$)	16.06	16.38	17.13
LSTM ($h = 256$)	16.08	16.37	17.12
LSTM ($h = 512$)	15.98	16.38	17.13
SPIN2	12.11	12.61	-
Structure-based models			
StructTrans	8.56	8.97	7.14
StructGNN	8.40	8.84	6.69
GCA	**7.09**	**7.49**	**6.06**

Table 1: Performance of different methods on CATH 4.2 dataset assessed by PERP (lower is better).

Though PERP matters from the perspective of natural language, REC that evaluates the ability of models in inferring sequences given determined structures is also crucial. We compare GCA with other structure-based models in Table 2.

Methods	Short	Single chain	All
StructTrans	31.59	30.35	33.90
StructGNN	30.90	30.85	35.25
GCA	**32.25**	**33.04**	**36.11**

Table 2: Performance of different methods on CATH 4.2 dataset assessed by REC (higher is better).

GCA obtains the highest REC on all three sets among these structure-based methods. Moreover, the recovery of StructGNN and StructTrans drops significantly in 'Short' and 'Single chain' sets, which suggests they are overfitting on long sequences and multiple chains, while GCA performs consistently well on them. As few structural features can be explored in short sequnce and single chain, the prediction is relatively difficult. However, the global information in GCA makes up for the deficiency of structural features of short chains, making performance significantly improved.

To compare with other methods, we conduct experiments on the standard TS50 dataset and show the results in Table 3. The methods for comparison include the CNN-based ProDCoNN (Zhang et al., 2020), the distance-map-based SPROF (Chen et al., 2019), the graph-based GVP (Jing et al., 2021) the sequential representation method SPIN (Li et al., 2014) and SPIN2 (O'Connell et al., 2018), the constraint satisfaction method ProteinSolver (Strokach et al., 2020), and the popular method Rosetta. GCA achieves remarkable performance and outperforms other methods by a large margin.

Methods	REC
Rosetta	30.0
SPIN	30.3
ProteinSolver	30.8
SPIN2	33.6
StructTrans	36.1
StructGNN	38.0
SPROF	39.2
ProDCoNN	40.7
GVP	44.1
GCA	**47.0**

Table 3: Performance of different methods on TS50 dataset assessed by REC (higher is better).

4 Conclusion

We introduce the consideration of global information and propose the global-context aware generative protein design method, consisting of local modules and global modules. The local module propagates neighborhood messages across layers, and the global module emphasizes long-term dependencies. Experimental results show that our proposed GCA method outperforms state-of-the-art methods on benchmark datasets. In 'Short' and 'Single chain' sets, the global-context aware mechanism significantly improves the performance, indicating the potentials to promote structure-based protein design.

5 Limitations

Though our proposed GCA method has taken the global context into consideration, there is still room for further improvement in efficiency. A possible solution is learning a global context vector for each protein, which we leave as future work.

References

TJ Brunette, Fabio Parmeggiani, Po-Ssu Huang, Gira Bhabha, Damian C Ekiert, Susan E Tsutakawa, Greg L Hura, John A Tainer, and David Baker. 2015. Exploring the repeat protein universe through computational protein design. *Nature*, 528(7583):580–584.

Yue Cao, Payel Das, Vijil Chenthamarakshan, Pin-Yu Chen, Igor Melnyk, and Yang Shen. 2021. Fold2seq: A joint sequence(1d)-fold(3d) embedding-based generative model for protein design. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1261–1271. PMLR.

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

Wenhao Gao, Sai Pooja Mahajan, Jeremias Sulam, and Jeffrey J. Gray. 2020. Deep learning in protein structural modeling and design. *Patterns*, 1(9):100142.

Po-Ssu Huang, Scott E Boyken, and David Baker. 2016. The coming of age of de novo protein design. *Nature*, 537(7620):320–327.

John Ingraham, Vikas Garg, Regina Barzilay, and Tommi Jaakkola. 2019. Generative models for graph-based protein design. In *Advances in Neural Information Processing Systems*, volume 32, pages 15820–15831. Curran Associates, Inc.

Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael John Lamarre Townshend, and Ron Dror. 2021. Learning from protein structure with geometric vector perceptrons. In *International Conference on Learning Representations*.

Andriy Kryshtafovych, Torsten Schwede, Maya Topf, Krzysztof Fidelis, and John Moult. 2019. Critical assessment of methods of protein structure prediction (casp)—round xiii. *Proteins: Structure, Function, and Bioinformatics*, 87(12):1011–1020.

Robert A Langan, Scott E Boyken, Andrew H Ng, Jennifer A Samson, Galen Dods, Alexandra M Westbrook, Taylor H Nguyen, Marc J Lajoie, Zibo Chen, Stephanie Berger, et al. 2019. De novo design of bioactive protein switches. *Nature*, 572(7768):205–210.

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

Ali Madani, Bryan McCann, Nikhil Naik, Nitish Shirish Keskar, Namrata Anand, Raphael R Eguchi, Po-Ssu Huang, and Richard Socher. 2020. Progen: Language modeling for protein generation. *arXiv preprint arXiv:2004.03497*.

James O'Connell, Zhixiu Li, Jack Hanson, Rhys Heffernan, James Lyons, Kuldip Paliwal, Abdollah Dehzangi, Yuedong Yang, and Yaoqi Zhou. 2018. Spin2: Predicting sequence profiles from protein structures using deep neural networks. *Proteins: Structure, Function, and Bioinformatics*, 86(6):629–633.

Leonardo FR Ribeiro, Yue Zhang, Claire Gardent, and Iryna Gurevych. 2020. Modeling global and local node contexts for text generation from knowledge graphs. *Transactions of the Association for Computational Linguistics*, 8:589–604.

David Sehnal, Sebastian Bittrich, Mandar Deshpande, Radka Svobodová, Karel Berka, Václav Bazgier, Sameer Velankar, Stephen K Burley, Jaroslav Koča, and Alexander S Rose. 2021. Mol* viewer: modern web app for 3d visualization and analysis of large biomolecular structures. *Nucleic Acids Research*.

Alexey Strokach, David Becerra, Carles Corbi-Verge, Albert Perez-Riba, and Philip M. Kim. 2020. Fast and flexible protein design using deep graph neural networks. *Cell Systems*, 11(4):402–411.e4.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

Kevin K Yang, Zachary Wu, and Frances H Arnold. 2019. Machine-learning-guided directed evolution for protein engineering. *Nature methods*, 16(8):687–694.

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