Self-Supervised Representation Learning of Protein Tertiary Structures (PtsRep): Protein Engineering as A Case Study

Junwen Luo[1]†, Yi Cai[1]†, Jialin Wu[1], Hongmin Cai[2], Xiaofeng Yang[1]* and Zhanglin Lin[1]*

[1]School of Biology and Biological Engineering, [2]School of Computer Science and Engineering, South China University of Technology, University Park, Guangzhou, Guangdong 510006, China.

* To whom correspondence should be addressed:

School of Biology and Biological Engineering, South China University of Technology, 382 East Outer Loop Road, University Park, Guangzhou, Guangdong 510006, China; Tel: +86 (20) 3938-0680; Fax: +86 (20) 3938-0601; Email: zhanglinlin@scut.edu.cn (Z.L.); biyangxf@scut.edu.cn (X.Y.).

† These authors contributed equally to this work.

Abstract

In recent years, deep learning has been increasingly used to decipher the relationships among protein sequence, structure, and function. Thus far deep learning of proteins has mostly utilized protein primary sequence information, while the vast amount of protein tertiary structural information remains unused. In this study, we devised a self-supervised representation learning framework to extract the fundamental features of unlabeled protein tertiary structures (PtsRep), and the embedded representations were transferred to two commonly recognized protein engineering tasks, protein stability and GFP fluorescence prediction. On both tasks, PtsRep significantly outperformed the two benchmark methods (UniRep and TAPE-BERT), which are based on protein primary sequences. Protein clustering analyses demonstrated that PtsRep can capture the structural signals in proteins. PtsRep reveals an avenue for general protein structural representation learning, and for exploring protein structural space for protein engineering and drug design.

Introduction

Self-supervised learning is a successful method for learning general representations from unlabeled samples[1-5]. In natural language processing (NLP), it takes the forms of Word2Vec (continuous skip-grow model, and continuous bag-of-words model)[1], next-token prediction[2], and masked-token prediction[3]. Recently these NLP-based techniques have been applied to representation learning of protein primary sequences[6-10]. Two representative examples are UniRep[10] which is based on next-token prediction, and a BERT[11] model (denoted as TAPE-BERT) which is based on masked-token prediction. Through protein transfer learning, both methods showed good performance for prediction of protein stability landscape, and green fluorescence protein (GFP) activity landscape. In these tasks, two datasets, one from Rocklin *et al*[12], and the other from Sarkisyan *et al*[13] were used. The former includes the sequences and chymotrypsin stability scores of more than 69,000 de novo designed proteins, natural proteins, and their mutants or other derivants. The latter contains more than 50,000 mutants of GFP from *Aequorea victoria*.

Structural biology has made the tertiary structures of many proteins accessible[14], and the advent of recent breakthrough of AlphaFold 2[15] will likely render protein structures more readily available. Thus far, representation learning of proteins has mostly processed protein primary sequence information, whereas the vast amount of protein tertiary structural information available in the PDB[16] database remains unused. How to utilize protein structural information in deep learning remains a fundamentally important question.

In this work, we present a self-supervised learning framework to learn embedded representations of protein tertiary structures (PtsRep) (Fig. 1). The learned representations summarized known protein structures into fixed-length vectors (738 dimensions). These were then transferred to three tasks: (1) protein fold classification, (2) prediction for protein stability, and (3) prediction for GFP fluorescence (Fig. 1C). We found that PtsRep performed well for fold reclassification of unlabeled structures, but most importantly, it significantly outperformed the benchmark methods (UniRep and TAPE-BERT) in the two protein engineering tasks. Thus, this self-supervised representation learning approach of protein structures provides an avenue to approximate or capture the fundamental features of protein structures, and has the promise to advance protein engineering and drug design.

Methods

1. Encoding of protein tertiary structures by *K* nearest residues

To encode the structural information for each residue in a given protein, we adopted a strategy from an algorithm which enables each residue to be represented by its $\boldsymbol{K}$ nearest residues (KNR) in the Euclidean space[17]. Each single residue was represented by its bulkiness[18], hydrophobicity[19], flexibility[20], relative spatial distance, relative sequential residue distance, and spatial position based on spherical coordinate system (Fig. 1A).

2. Bidirectional language model for self-supervised representation learning of protein structures

Given a protein of $\boldsymbol{n}$ residues $(\boldsymbol{w_1}, \boldsymbol{w_2}, \dots, \boldsymbol{w_n})$, for each residue, $\boldsymbol{p}$ is the probability[21] of predicted residue corresponding to the actual residue, and the probability for the sequence was calculated by modeling the probability of the residue at position $\boldsymbol{t}$ $(\boldsymbol{w_t})$, for the given history[22] $(\boldsymbol{w_1}, \boldsymbol{w_2}, \dots, \boldsymbol{w_{t-1}})$:

$$\boldsymbol{p}(\boldsymbol{w_1}, \boldsymbol{w_2}, \dots, \boldsymbol{w_n}) = \prod_{\boldsymbol{t=1}}^{\boldsymbol{n}} \boldsymbol{p}(\boldsymbol{w_t} \mid \boldsymbol{w_1}, \boldsymbol{w_2}, \dots, \boldsymbol{w_{t-1}})$$

At each position $\boldsymbol{t}$, the forward long short-term memory (LSTM) layer output a context-dependent representation $\boldsymbol{h_t^{\rightarrow}}$, which was used to predict the next residue $\boldsymbol{w_{t+1}}$ with a Softmax layer[21]. We additionally trained the reverse LSTM to provide a bidirectional context for each residue, which produced the representation $\boldsymbol{h_t^{\leftarrow}}$ for $\boldsymbol{w_t}$, for the given the history $(\boldsymbol{w_{t+1}}, \boldsymbol{w_{t+2}}, \dots, \boldsymbol{w_n})$.

$$p(w_1, w_2, \dots, w_n) = \prod_{t=1}^{n} p(w_t \mid w_{t+1}, w_{t+2}, \dots, w_n)$$

Our formulation maximized the sum of the log likelihoods in both the forward and backward directions as follows:

$$S = \sum_{t=1}^{n} (log\ p(w_t \mid w_1, w_2, \dots, w_{t-1};\ \theta_w, \theta_{LSTM}^{\rightarrow}, \theta_s) + log\ p(w_t \mid w_{t+1}, w_{t+2}, \dots, w_n;\ \theta_w, \theta_{LSTM}^{\leftarrow}, \theta_s))$$

Where, we tied the parameters for the residue representation (θ_w), LSTM layer (θ_{LSTM}) and the Softmax layer (θ_s) in both the forward and backward directions. At each position t, only the history information was given to avoid leakage of subsequent sequence position information[23].

We used the cross entropy as the loss function[21]. Then we defined d as the distance from the current position t, and R(t, d) or the sum of log likelihood in both the forward and backward directions as follows:

$$R(t, d) = log\ p(w_t \mid w_1, w_2, \dots, w_{t-d}; \theta_w, \theta_{LSTM}^{\rightarrow}, \theta_s)$$
$$+ log\ p(w_t \mid w_{t+d}, w_{t+d+1}, \dots, w_n; \theta_w, \theta_{LSTM}^{\leftarrow}, \theta_s)$$

Similarly, we tied the parameters for the residue representation (θ_w), LSTM layer (θ_{LSTM}) and Softmax layer (θ_s) in both the forward and backward directions. Considering that immediately adjacent amino acid residues may be easy to identify with the input structural information, we omitted these two residues in both directions. The final loss was the average of calculated $Loss$ for all position of t:

$$Loss(n, d) = \frac{1}{4n - 4d - 2} \sum_{t=1}^{n} \sum_{r=d}^{d+1} R(t, r)$$

3. Additional training details

We chose 768 as the dimension of embedding, which followed the Bidirectional Encoder Representation from Transformers (BERT)[3]. Because the length of each protein sequence was different, we used a batch size of 1 with a constant learning rate 2e-3. Swish[24] activation was applied to each layer as follows:

$$\boldsymbol{Swish}(\boldsymbol{x}) = \frac{\boldsymbol{x}}{\mathbf{1} + \boldsymbol{e}^{\boldsymbol{x}}}$$

Considering that the activations of each layer had a different distribution, a layer normalization[25] was applied to each layer.

For these stacked networks, the optimizer Adam[26] was selected. The model was trained for 2.5 million weight updates corresponding to ~1 epoch. Convergence was determined as no improvement in the validation contrastive loss for 5 epochs.

4. Datasets

1) Self-supervised learning dataset.

We used ProteinNet[27] as the training and validation set of the self-supervised model. We first collected the 90% thinning version of ProteinNet12[27], which had 49,600 protein chains with tertiary structures. Then we excluded the protein chains (i) which have missing Cα coordinates, or (ii) which are longer than 700 aa or shorter than 30 aa. The resulting dataset had 35,568 protein chains. We used 95% of the data as the training dataset, and 5% as the validation dataset.

2) Stability landscape prediction dataset

We used the dataset from Rocklin et al[12]. This set includes the sequences and chymotrypsin stability scores of 69,034 proteins (17,773 de novo designed proteins, 10,674 mutants of these

designed proteins, 1,193 natural proteins, 2,423 mutants of these natural proteins, 24,900 scrambled versions of the proteins, and 12,071 control inactivated sequences where a buried aspartate residue was inserted) (Supplementary Table 1). Among these proteins or mutants, we used a total of 16,431 proteins or mutants with structural information (16,159 de novo designed mini proteins and 272 natural proteins) for training and validation, and 12,851 point mutants of 14 de novo designed proteins and 3 natural proteins with structural information for testing[12]. For training and validation, we used an 80%/20% splitting strategy.

3) Fluorescence landscape prediction

We used the dataset from Sarkisyan et al[13], which contains more than 50,000 mutants of green fluorescent protein from *Aequorea victoria*. The number of mutations in the mutants were from 1 to 15[28]. Following a previous study, we used the mutants with 1~3 mutations for training and validation, and used the mutants with 4 or more mutations for testing[15]. For training and validation, we used an 80%/20% splitting strategy (Supplementary Table 1).

4) Benchmark representations

We used UniRep, which is an mLSTM model trained on about 24 million protein sequences, as a benchmark representation[10]. We downloaded the trained weights and obtained the representations using the code in https://github.com/churchlab/UniRep. We chose the 1900-dimensional UniRep representation, which was reported to perform best among the different dimensions[10].

We also used TAPE-BERT trained on about 32 million protein sequences as a second benchmark representation[3]. We downloaded the trained weights and obtain the representation using the code in https://github.com/songlab-cal/tape.

Results

1. How PtsRep learned the representations of protein tertiary structures

The framework used in our work consists of three steps as shown in Fig. 1. (A) To feed the protein tertiary structural information, we adopted an algorithm[17] which enables each residue to be represented by the properties of its *K* nearest residues (KNR) in the Euclidean space[17]. We had varied *K*, and set it at 15 after optimization (data not shown). A dataset of 35,568 protein chains were selected from PDB, with 95% used for training, 5% for validation. (B) To learned the structural representations, we first used Bi-LSTM to extract embedded features of protein tertiary structures, and then iterated the subnetwork by maximizing the accuracy of predicting the four nearest noncontiguous residues at each amino acid residue of a given protein structure, *i.e.*, positions -3, -2, +2, +3 (the present residue is defined as position 0), after preliminary optimization (data not shown). This yielded an optimal model, which summarized any protein tertiary structures into 768-dimensional vectors of the protein sequence length. (C) Finally, this trained PtsRep was applied to protein engineering tasks through a downstream network, as shown in Fig. 1C. To better align with the benchmark TAPE-BERT[11], we used the same linear regression model.

2. PtsRep performed well for protein fold classification

To examine what PtsRep learned from the PDB dataset, we used t-distributed stochastic neighbor embedding (t-SNE)[29] to test its unsupervised clustering ability of 15,444 semantically related but unlabeled proteins, which have been classified by the Structural Classification of

Proteins (SCOP)[30]. We found that PtsRep separated the five different types of proteins well (Fig. 2, right column). Using the Davies-Bouldin Index[31] (DBI), a general index for evaluating clustering algorithms, we found that the DBI for PtsRep was only 1.27. In contrast, for the baseline representation KNR which utilized the same amount of structural information but without the training of PtsRep, the DBI was 11.62 (Fig. 2, left column). Therefore, the trained PtsRep model improved the clustering by 9.2 times in this SCOP classification task.

3. PtsRep outperformed benchmarks on protein stability prediction

We then used the downstream tasks as shown in Fig. 1C to further evaluate the protein structural embeddings learned by PtsRep. As shown in Table 1, we compared Spearman's ρ, accuracy (ACC), MSE. Protein stability is a key point of protein engineering that has influence on the production yields[32] and reaction rates[33]. In this study, the dataset from Rocklin et al[12] consists of the following four protein topological structures: $\alpha\alpha\alpha, \alpha\beta\beta\alpha, \beta\alpha\beta\beta, \beta\beta\alpha\beta\beta$, where α denotes a helix and β denotes a sheet. It should be noted that the training/validation dataset for the benchmarks UniRep and TAPE-BERT contains 56,180 proteins (Supplementary Table 1), among which only 16,431 with structural information were suitable for the PtsRep in this study. Thus, PtsRep was trained on only 29.3% of the data used for UniRep and TAPE-BERT. However, all the proteins in the test dataset for the benchmarks have solved structures, thus the same test dataset was used for both UniRep and TAPE-BERT, and PstRep in this study.

We found that PtsRep significantly outperformed the best benchmark TAPE-BERT (Spearman's ρ = 0.79 *versus* 0.73), and much more so against the baseline KNR (Spearman's ρ =

0.79 *versus* 0.37). PtsRep also showed generally better performance on accuracy and MSE, compared with the benchmarks. Interestingly, the benchmark UniRep performed diffidently for different protein topologies (e.g., $\alpha\beta\beta\alpha$, $\alpha\alpha\alpha$), but PtsRep performed more evenly in this aspect, with one of the smallest standard deviations among all the methods, which was significantly better compared with the benchmark UniRep (0.11 *versus* 0.25), but was similar to that of BERT (0.11 *versus* 0.08) (Supplementary Table 2).

4. PtsRep improved the efficiency of green fluorescent protein engineering

Fluorescent proteins are useful in conveniently assessing the protein fitness landscapes in terms of activity[13]. As described in the previous study[11] using the GFP mutant dataset of Sarkisyan *et al*[13], we allocated the mutants with 1~3 mutations as the dataset for training and validation, and the mutants with 4 or more mutations as the dataset for testing. As shown in Table 1, PtsRep outperformed the benchmarks for the test dataset (Spearman's ρ = 0.70 *versus* 0.67).

We wanted to explore what these metrics mean for the actual practice of protein engineering. To this end, we directly probed how much "testing budget" PtsRep would take to arrive at (or "prioritize") target mutants. We selected the top 0.1% of the GFP mutants in the test dataset in terms of brightness, or 26 mutants in total (with 4 to 15 mutations), and tested the ability of each method to prioritize this set of brightest GFP under different test budgets. We trained one-hot, KNR representations as the baselines. As shown in Fig. 3A, PtsRep performed better than the benchmarks UniRep and BERT, and significantly better than the baselines. For example, PtsRep

achieved a 70% recall rate for this top 0.1% mutants with about 380 mutants searched, compared with 559 for BERT, 737 for UniRep, and 1,609 for the baseline KNR (Fig. 3A).

Subsequently, we probed how many trials PtsRep would take to identify the single brightest GFP protein. As shown in Fig. 3B, we found that the brightest protein was found at the 28th trial for PtsRep. The next best approach arrived at around the 70th trial, or at a cost of 150% more. It took 497 trials for the baseline KNR.

We again used t-SNE to test the ability of PtsRep to cluster the GFP mutants. As shown in Fig. 4, for PtsRep the mutants with bright fluorescence were more clustered with a DBI index of 2.53, compared to KNR with a more scattered distribution with a DBI of 4.62 (Supplementary Table 3).

Discussion

In this work, we presented a self-supervised learning-based framework to learn general representations from unlabeled protein tertiary structures in the PDB database (PtsRep), and transferred the learned model for two presently well recognized protein engineering benchmark tasks: predictions of protein stability and GFP fluorescence, based on two commonly available datasets. PtsRep enabled outstanding and partially superior performances on both tasks, compared with the protein sequences-based benchmark methods UniRep and TAPE-BERT.

As far as protein engineering is concerned, while our study showed that one can extrapolate results from mutants of 1-3 mutations for mutants of 4-15 mutations, an optimal experimental search strategy remains to be further defined, for using deep learning to reduce the experimental burden for a better practice of protein engineering. It is also possible to combine PstRep with simulated annealing[34] and Bayesian optimization[35], and use PstRep for disease diagnosis, by estimating the effects of nonsynonymous single-nucleotide polymorphisms[10,36,37].

While what deep learning learns from data is often considered a black box, we strived to partially dissect what PtsRep has learned from the PDB database. We surmised that PtsRep has at least understood which protein structures are stable, since unstable protein structures would not have appeared in the PDB database, since unstable protein structures lead to inactive protein mutants. We also know from the practice of directed evolution of enzymes that about 30% of randomly mutated protein variants are inactive[38]. This explains in part the improvement we saw for the better performance of PtsRep as shown in Fig. 3. This is also consistent with the fact that

PtsRep is able to perform much better than KNR, in terms of protein fold classification, as shown in Fig. 2.

However, what remains to be understood is how PtsRep is able to better cluster GFP variants with higher activities (Fig. 4). A similar observation was found for TAPE-BERT, which is based on protein sequences. It is noteworthy that the KNR format input for PtsRep in this study contains protein primary sequence information, thus it is likely that the better clustering of GFP seen for PtsRep was also sequence-based.

Last but not the least, we wish to point out that, comparing to the best benchmark TAPE-BERT, which used semi-supervised learning with 96 million parameters to train a deep contextual language model on 32 million sequences, our model was built with only 2.6 % of the parameters, and trained on only 0.11% of the protein entries (*i.e.*, those with structural information).

Acknowledgments

This work was supported by the National Key R&D Program of China (2018YFA0901000), and the Guangzhou Science and Technology Program key projects (201904020016). We thank Xing Zhang for technical assistance.

References

1 Mikolov T, Chen K, Corrado G *et al*. Efficient estimation of word representations in vector space. *arXiv*, 1301.3781 (2013).

2 Peters M E, Neumann M, Iyyer M *et al.* Deep contextualized word representations. *arXiv:1802.05365* (2018).

3 Devlin J, Chang M-W, Lee K *et al.* BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv*, 1810.04805 (2019).

4 Gidaris S, Singh P, Komodakis N. Unsupervised representation learning by predicting image rotations. *arXiv*,1803.07728 (2018).

5 Goyal P, Mahajan D, Gupta A *et al*. Scaling and Benchmarking Self-Supervised Visual Representation Learning. *International Conference on Computer Vision*, 6400-6409 (2019).

6 Heinzinger M, Elnaggar A, Wang Y *et al.* Modeling aspects of the language of life through transfer-learning protein sequences. *BMC Bioinformatics* **20**, 723 (2019).

7 Yang K K, Wu Z, Bedbrook C N *et al*. Learned protein embeddings for machine learning. *Bioinformatics* **34**, 2642-2648 (2018).

8 Rives A, Meier J, Sercu T *et al.* Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *bioRxiv*, 622803 (2020).

9 Bepler T, Berger B. Learning protein sequence embeddings using information from structure. *arXiv*,1902.08661 (2019).

10 Alley E C, Khimulya G, Biswas S *et al.* Unified rational protein engineering with

sequence-based deep representation learning. *Nature Methods* **16**, 1315-1322 (2019).

11 Rao R S, Bhattacharya N, Thomas N *et al.* Evaluating protein transfer learning with TAPE. *Advances in Neural Information Processing Systems* **32**, 9689-9701 (2019).

12 Rocklin G J, Chidyausiku T M, Goreshnik I *et al*. Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science* **357**, 168-174 (2017).

13 Sarkisyan K S, Bolotin D A, Meer M V *et al*. Local fitness landscape of the green fluorescent protein. *Nature* **533**, 397-401 (2016).

14 Jin P, Bulkley D, Guo Y *et al.* Electron cryo-microscopy structure of the mechanotransduction channel NOMPC. *Nature* **547**, 118-122 (2017).

15 Jumper J, Evans R, Pritzel A *et al.* High accuracy protein structure prediction using deep learning. *Fourteenth Critical Assessment of Techniques for Protein Structure Prediction (Abstract Book)*, 22-24 (2020).

16 Berman H M, Kleywegt G J, Nakamura H *et al.* The protein data bank archive as an open data resource. *Journal of Computer-aided Molecular Design* **28**, 1009-1014 (2014).

17 Wang J, Cao H, Zhang J Z H *et al.* Computational Protein Design with Deep Learning Neural Networks. *Scientific Reports* **8**, 6349 (2018).

18 Zimmerman J M, Eliezer N, Simha R. The characterization of amino acid sequences in proteins by statistical methods. *Journal of Theoretical Biology* **21**, 170-201 (1968).

19 Kyte J, Doolittle R F. A simple method for displaying the hydropathic character of a protein. *Journal of Molecular Biology* **157**, 105-132 (1982).

20 Huang F, Nau W M. A conformational flexibility scale for amino acids in peptides. *Angewandte Chemie International Edition* **42**, 2269-2272 (2003).

21 Bengio Y, Ducharme R, Vincent P *et al.* A neural probabilistic language model. *Journal of Machine Learning Research* **3**, 1137-1155 (2003).

22 Jozefowicz R, Vinyals O, Schuster M *et al.* Exploring the limits of language modeling. *arXiv*, 1602.02410 (2016).

23 Peters M E, Ammar W, Bhagavatula C *et al.* Semi-supervised sequence tagging with bidirectional language models. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 1756-1765 (2017).

24 Ramachandran P, Zoph B, Le Q V. Searching for activation functions. *arXiv*, 1710.05941 (2017).

25 Ba J L, Kiros J R, Hinton G E. Layer normalization. *arXiv*, 1607.06450 (2016).

26 Kingma D P, Ba J L. Adam: A method for stochastic optimization. *arXiv*, 1412.6980 (2015).

27 Mohammed A. ProteinNet: a standardized data set for machine learning of protein structure. *BMC Bioinformatics* **20**, 311 (2019).

28 Huang P S, Boyken S E, Baker D. The coming of age of de novo protein design. *Nature* **537**, 320-327 (2016).

29 van der Maaten L, Hinton G. Visualizing Data using t-SNE. *Journal of Machine Learning Research* **9**, 2579-2605 (2008).

30 Fox N K, Brenner S E, Chandonia J M. SCOPe: Structural Classification of Proteins -

extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Research* **42**, D304-309 (2014).

31 Davies D L, Bouldin D W. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **1**, 224-227 (1979).

32 Kwon W S, Da Silva N A, Kellis J T, Jr. Relationship between thermal stability, degradation rate and expression yield of barnase variants in the periplasm of Escherichia coli. *Protein Engineering, Design and Selection* **9**, 1197-1202 (1996).

33 Bommarius A S, Paye M F. Stabilizing biocatalysts. *Chemical Society Reviews* **42**, 6534-6565 (2013).

34 Biswas S, Kuznetsov G, Ogden P J *et al.* Toward machine-guided design of proteins. *bioRxiv*, 337154 (2018).

35 Yang K K, Chen Y X, Lee A *et al.* Batched stochastic Bayesian optimization via combinatorial constraints design. *International Conference on Artificial Intelligence and Statistics* **89** (2019).

36 Fang J. A critical review of five machine learning-based algorithms for predicting protein stability changes upon mutation. *Briefings in Bioinformatics* **21**, 1285-1292 (2020).

37 Li M, Kales S C, Ma K *et al.* Balancing protein stability and activity in cancer: a new approach for identifying driver mutations affecting CBL ubiquitin ligase activation. *Cancer research* **76**, 561-571 (2016).

38 Romero P A, Arnold F H. Exploring protein fitness landscapes by directed evolution. *Nature*

Reviews Molecular Cell Biology **10**, 866-876 (2009).

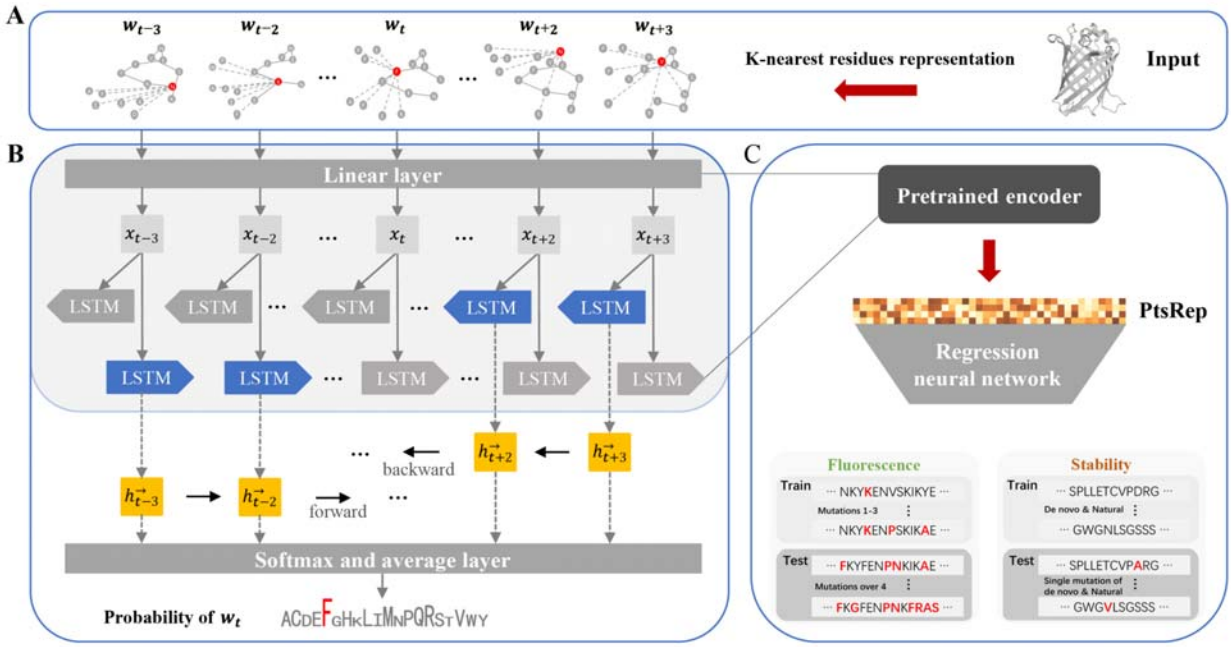

Figure 1. Workflow of PtsRep for learning and applying protein tertiary structure representations. (A) Protein structures from PDB were encoded with the KNR (K nearest residues) algorithm, with each amino acid represented by the nearest 15 amino acids and their features. (B) PtsRep was trained to perform contextual noncontiguous residue prediction using a cross-entropy loss function, and internally represent proteins. (C) A regression neural network was used to transfer the embedded representations for downstream protein engineering tasks.

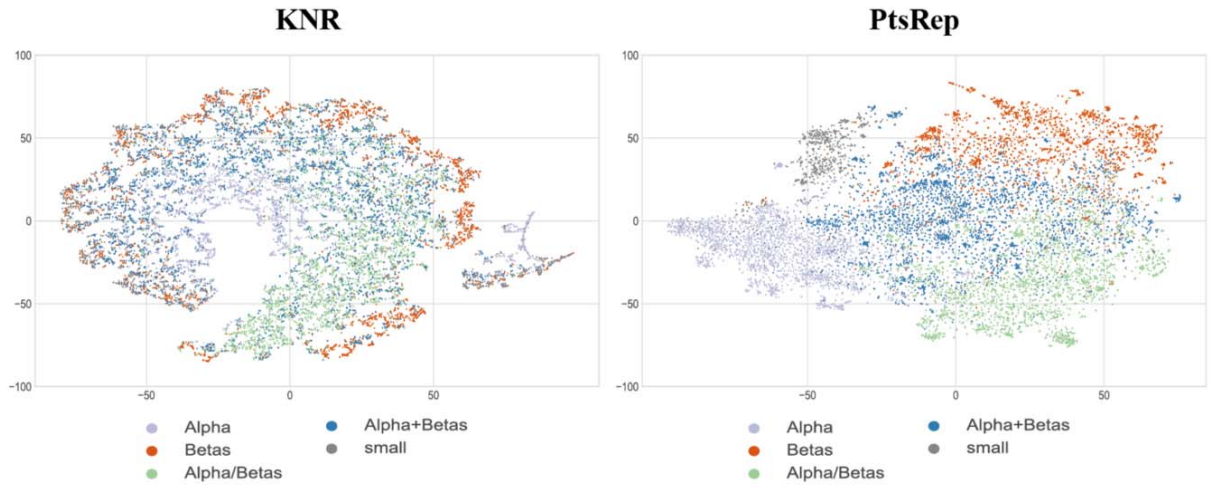

Figure 2. t-SNE (t-distributed stochastic neighbor embedding) representations of KNR (left) and PtsRep (right) for 15,444 proteins classified by the Structural Classification of Proteins (SCOP)[30], respectively. t-SNE projections from embedded space onto a low dimensional representation are shown, which represent sequences from SCOP colored by ground-truth structural classes of five types: alpha, beta, alpha/beta, alpha+beta, and small proteins.

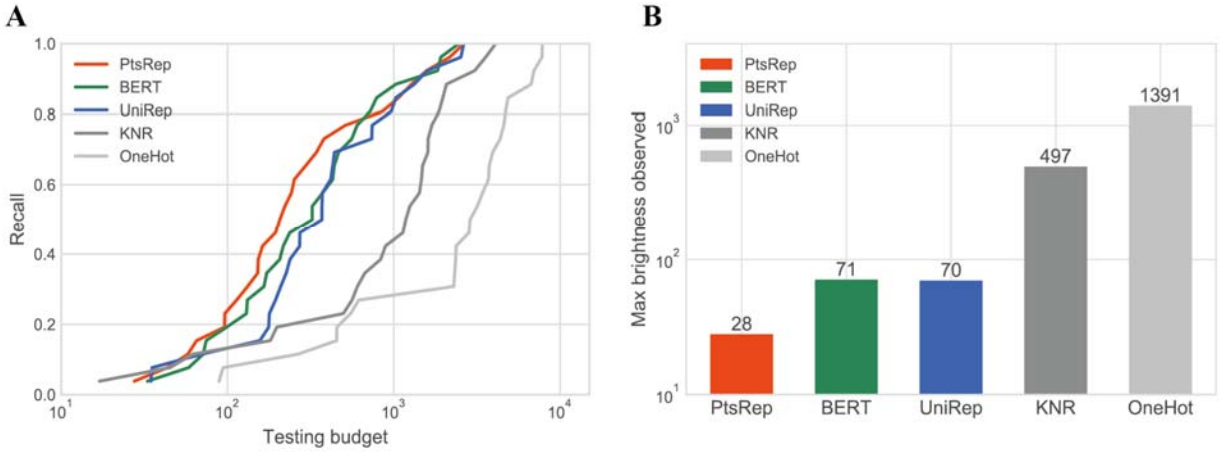

Figure 3. (A) The brightest 0.1% of GFP mutants observed *versus* the testing budget for each representation method. The abscissa uses an exponential scale. (B) The trials taken to identify the brightest GFP mutant for each representation method.

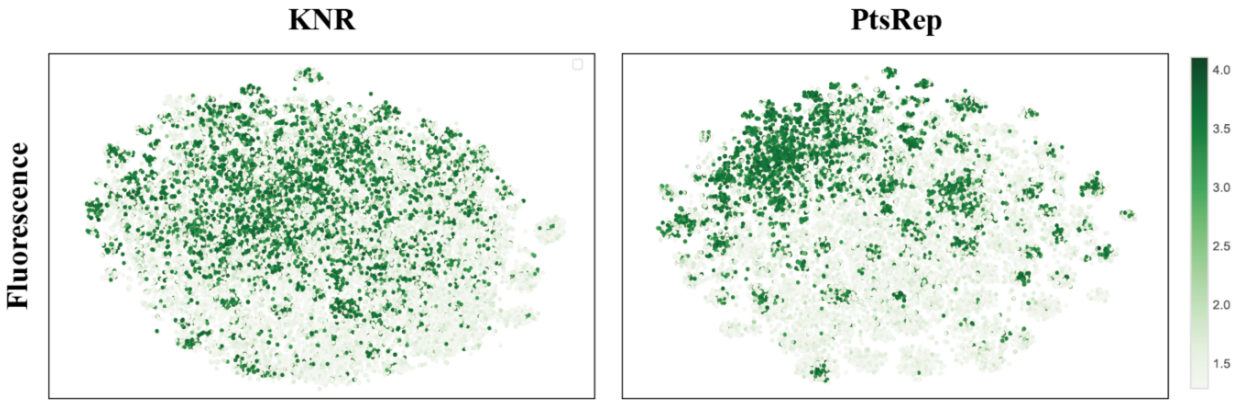

Figure 4. t-SNE visualization of KNR (left) and PtsRep (right) for GFP mutants with 4 to 15 mutations, respectively.

Table 1. Embedding performances of downstream tasks, using outlined benchmarks or baselines.

Method	Param	Data	Stability			Fluorescence		
			ρ	ACC	MSE	ρ	ACC	MSE
UniRep[10]	182M	24M	0.73∗	0.69∗	0.15	0.67∗	0.96	**0.20**∗
BERT[11]	92M	32M	0.73∗	0.70∗	0.12	0.68∗	0.96	0.22∗
Bepler[9]	19M	0.02M	0.64∗	0.67∗	-	0.33∗	-	2.17∗
One-Hot	0	32M	0.19∗	0.58∗	0.70	0.14∗	0.86	2.69∗
KNR	0	0.04M	0.37	0.71	0.38	0.67	0.92	0.66
PtsRep	2.5M	0.04M	**0.79**	**0.73**	**0.09**	**0.70**	**0.97**	0.57

* taken from ref [11].