Received: 19 August 2019 | Revised: 13 December 2019 | Accepted: 14 December 2019

DOI: 10.1002/prot.25868

RESEARCH ARTICLE

PROTEINS WILEY

ProDCoNN: Protein design using a convolutional neural network

Yuan Zhang[1] | Yang Chen[1] | Chenran Wang[1] | Chun-Chao Lo[1] | Xiuwen Liu[2] | Wei Wu[1] | Jinfeng Zhang[1]

[1]Department of Statistic, Florida State University, Tallahassee, Florida

[2]Department of Computer Science, Florida State University, Tallahassee, Florida

Correspondence
Jinfeng Zhang, Department of Statistic, Florida State University, Tallahassee, FL 32306.
Email: jinfeng@stat.fsu.edu

Funding information
National Institutes of Health, Grant/Award Number: R01GM126558

Peer Review
The peer review history for this article is available at https://publons.com/publon/10.1002/prot.25868.

Abstract

Designing protein sequences that fold to a given three-dimensional (3D) structure has long been a challenging problem in computational structural biology with significant theoretical and practical implications. In this study, we first formulated this problem as predicting the residue type given the 3D structural environment around the C_α atom of a residue, which is repeated for each residue of a protein. We designed a nine-layer 3D deep convolutional neural network (CNN) that takes as input a gridded box with the atomic coordinates and types around a residue. Several CNN layers were designed to capture structure information at different scales, such as bond lengths, bond angles, torsion angles, and secondary structures. Trained on a very large number of protein structures, the method, called ProDCoNN (protein design with CNN), achieved state-of-the-art performance when tested on large numbers of test proteins and benchmark datasets.

KEYWORDS

convolutional neural network, inverse folding problem, ProDCoNN, protein design, protein engineering

1 | INTRODUCTION

As "machines of life," proteins play crucial roles in almost all cellular processes, such as transcription, translation, signaling, and cell cycle control. Computational protein design (CPD) is a technique by which proteins are computationally designed for specific structure and/or functions. Significant progress in CPD has been made in the past few decades[1] such as enzymes design[2-8] and membrane protein design.[9,10] There are several different sub-problems in CPD. The inverse protein folding (IPF) problem, which was raised in 1983 by Pabo,[11] is the problem of finding the amino acid sequences that fold into a known three-dimensional (3D) structure. Solving this problem will improve our fundamental understanding of the sequence-structure relationship of proteins. Another problem with wide applications is to design proteins with new functions, such as enzyme design.[3,4,7,12] The design will usually start from a known structure, which will be kept in the process, and modify part of the sequence to achieve a new function. Variations of the problem include specificity design and affinity design, where the functions of designed proteins are not new, but some properties of functions are the target of CPD. The third type of CPD is to change the chemical or physical properties of the proteins, such as improvements in the stability of the designed proteins.[13] Increasing the thermostability of enzymes can be very useful in manufacturing industry, where enzymes are used for production of chemicals.

In this study we tackle mainly the first problem: designing protein sequences that fold to a known 3D structure—the IPF problem. A variation of our model can also be used to predict single residue mutations that stabilize a given protein structure.

There have been some remarkable successes in IPF in the past. In 1997, Stephen Mayo and coworkers successfully designed the first protein completely de novo.[14] In 1995, Desjarlais and Handel designed a computational framework for the de novo design of hydrophobic cores.[15] Raha and coworkers developed the sequence

prediction algorithm (SPA) in 2000 to design protein sequences that can fold to a given 3D backbone structure.[16] The agreement between the predicted sequences and the native sequence of each backbone template ranges from 24% to 28% on four protein superfamily motifs: SH3 domain, the homeodomain (HM), the fibronectin type III (FNIII) domain, and the RNA recognition motif (RRM). In 2000, Kuhlman and Baker introduced RosettaDesign[17] to identify low energy amino acid sequences for target protein structures. Later in 2003, it was used to redesign nine naturally occurring proteins, and the redesigned sequences were on average 35% identical to the wild-type sequence.[18] Hu and coworkers developed a nonlinear scoring function for selecting sequences comparable with a given structure.[19] Most of the above methods used energy-based method, which starts with random protein sequences and iteratively optimizes an energy function via mutations until the energy score reaches a minimum. Another approach is using local fragment structures from a target structure, which is compared to the fragment library of known protein structures.[20-22] In 2010, Liang Dai et al introduced RosettaDesign-SR, which uses a sequence profile derived from the sequences of five residue fragments in a fragment library.[23] The average sequence identity compare to wild-type sequence is 35% over 100 sequences designed for each protein for 33 training proteins.

Calculating the sequence identity to the wild-type (or native) sequence (also called recovery rate of the native sequence for a given structure) is a common assessment method for the designed sequences.[17,24] The average sequence recovery rates achieved by the current top-performing protein-design programs are around 35% on a small number of proteins. For the latest review articles on CPD and IPF problem see References 1, 25, 26.

The number of protein sequences and structures in the Protein Data Bank (PDB) has grown significantly in recent years. As of November 2018, the PDB[27,28] contains 135 685 protein structures. The increase of data allows more complex models to be built for protein design. In the past decade, research in deep neural networks (DNNs or deep learning methods) has made very rapid progress and provided the best solutions to many problems such as image recognition,[29-32] speech recognition,[33-37] and natural language processing[38-42] by training complex models using very large volumes of training data. Unlike conventional machine learning techniques that require a feature extraction step to transform the raw data into a suitable representation as the input, DNNs allow raw data to be fed directly to a network with certain architecture.[29,43,44] In deep learning, more efforts are made in selecting the type of DNNs and designing specific architecture for the selected DNN framework. In the past few years, deep learning has seen its applications in computational studies of protein structures, such as protein secondary structure prediction,[45-49] protein contact map prediction,[50-53] and protein–protein interaction prediction.[54-56]

DNNs have also been used for protein design. In 2014, Zhou and co-workers developed SPIN (sequence profiles by integrated neural network) based on fragment-derived sequence profiles and structure-derived energy profiles.[54,55,57] Both local and nonlocal features were designed and served as input for a two hidden layer neural network, which contained 51 hidden neurons and one bias. SPIN yielded an average sequence recovery of 30.7% for a dataset with 1532 proteins. Later in 2018, SPIN was upgraded to SPIN2 with more local features including two more backbone angles as well as more nonlocal features such as contact numbers.[58] This network included three hidden layers with 500 sigmoid nodes each. The output layer had 20 SoftMax nodes representing the 20 types of amino acid residues. SPIN2 achieved an average sequence recovery of 34.4% for the dataset with 1532 proteins. Another study adopting a DNN for protein design, conducted by Wang et al in 2018,[59] used structure features as input, such as cosine and sine values of backbone dihedrals, the total solvent accessible surface area of backbone atoms, atom distances, number of hydrogen bonds and secondary structures. A multilayer neural network was constructed including a residue probability network and a weight-network followed by several fully-connected layers, and a 20-dimensional SoftMax layer is used as the final output. In this study, the best recovery rate was 34% on a dataset with 10 173 proteins (30% sequence identity), and 38.3% on a dataset with 17 607 proteins (90% sequence identity).

Among all the deep learning frameworks, convolutional neural network (CNN)[60-62] has been widely used, especially for object recognition. The IPF problem can be formulated as a 3D object recognition problem: predicting the residue type given the 3D environment—the backbone structure of proteins. The key structure information determining the residue type at a particular position on a given protein structure is likely the atoms surrounding this residue, as shown in Figure 1A. We used a gridded box centered on the residue to get the local 3D environment information, which consists of the coordinates and types of all the atoms in the box. Now the problem becomes predicting the residue type given a point cloud—coordinates and types of all the atoms in the box (called the Box). To that end, we designed a CNN architecture including three parallel layers which are used to capture local protein structure features, such as bonds, angles, and dihedral angles, followed by another layer recognizing the secondary structure of the proteins (Figure 1B). The Box also contains atom-atom contact information naturally. Instead of extracting structural features as other studies did, we fed the Box directly to the CNN to learn all the parameters in the CNN model for residue type prediction. We used a sliding window going through the sequence and predicted each residue type one-by-one. Compared with SPIN and Wang et al's model (Wang's model), our method, called ProDCoNN (Protein Design using CNNs), has achieved substantially improved performance (see Section 3 for details).

2 | MATERIALS AND METHODS

Our method tackles protein design problem by predicting one residue, called target residue, at a time using the local structural information surrounding the target residue. We use a gridded box centered on the target residue to capture the local structural information around it, as shown in Figure 1. The cubic box of edge length 18 Å is centered on the C_α atom of the target residue. The box is gridded with each voxel being unit size (1 Å × 1 Å × 1 Å). With this resolution (each box has 5832 voxels), most of the voxels contains no more than one heavy atom. For each target residue, the protein is rotated to make its C_α—C bond lying along the *x*-axis, and its C_α—N bond lying on the *x*-*y* plane.

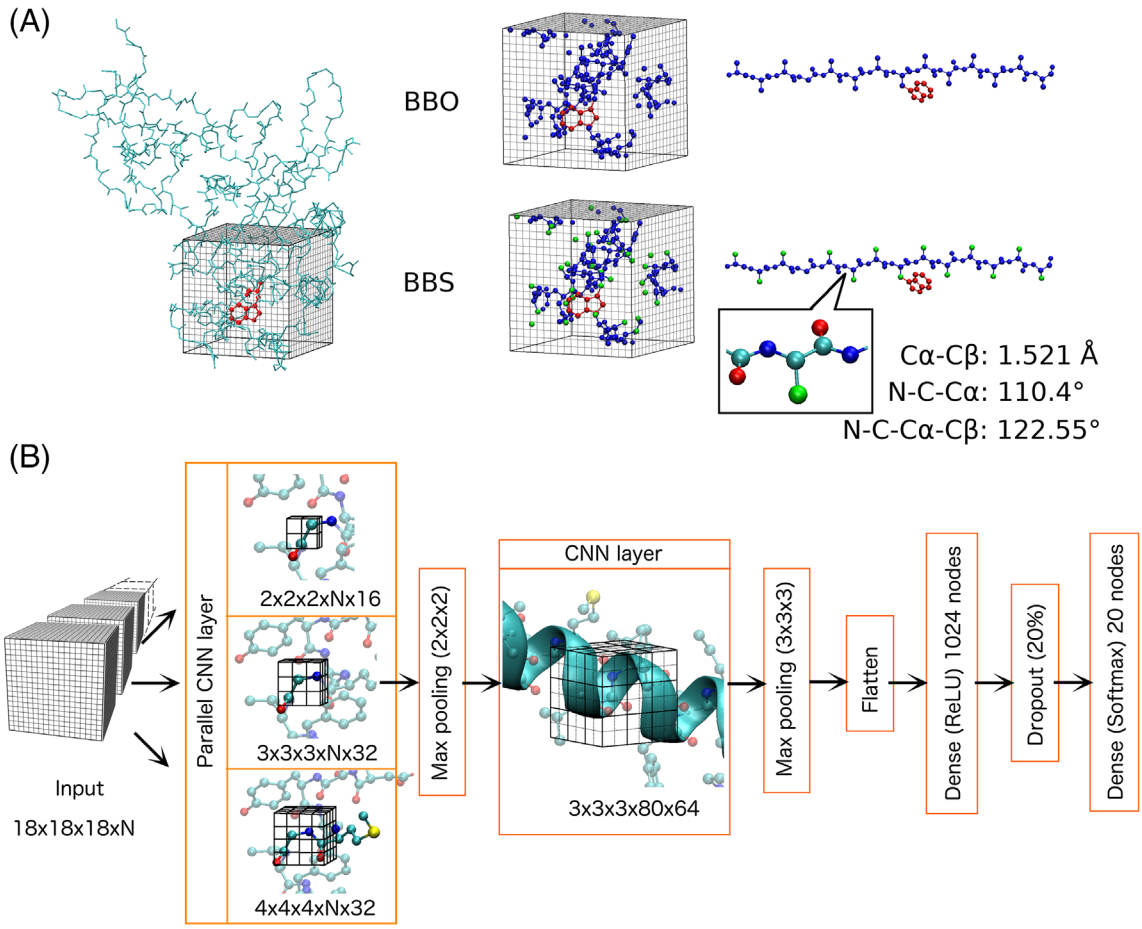

FIGURE 1 Architecture of neural network. A, Left: snapshot of the gridded box that captures the environment atoms to predict the amino acid type (red). Right: visualization of atoms (blue) captured by the box which used as input to CNN sequence prediction. BBO model uses the backbone atoms (C, Cα, N, O, and the extra oxygen atom OXT on the terminal carboxyl group) and pseudo-C_β atom which added manually with the bond length of C_α—C_β as 1.521 Å, bond angle N—C—C_α as 110.4° and dihedral angle N—C—C_α—C_β as 122.55°. BBS model uses the same atom set but label C_β (green) on nontarget residues differently based on the residue types in the protein sequence. B, The architecture of the designed deep Neural Network. The input consists of N ($N_{channel}$) set of 18× 18× 18 gridded boxes, and each channel represents the occupation of one type of atom

Each atom type is represented by a different channel (analogous to RGB color channels in images). With N types of heavy atoms, the dimension of input data is $18 \times 18 \times 18 \times N$. If an atom is only represented by the voxel it occupies, the exact coordinate information of the atom is lost. To overcome this limitation caused by the discretization of the 3D space around the target residue, we smooth the input data using 3D truncated Gaussian functions with SD $\sigma_x = \sigma_y = \sigma_z = r$, where r is the Van der Waals radius of the atom in a given voxel. The values for the radius are 0.7, 0.65, and 0.6 Å for carbon, nitrogen, and oxygen, respectively. Each heavy atom is represented by a Gaussian function, whose density is spread over the voxel the atom occupies and the 26 adjacent voxels. The exact densities in each voxel are dependent on the coordinate of the atom. All the side chain atoms are removed from the input data.

In addition to keeping all the backbone atoms (C, C_α, N, and O) and the extra oxygen atom OXT on the terminal carboxyl group, we also added a pseudo-C_β atom to each residue including GLY. The C_β atom at residue i has bond length for C_α^i—C_β^i bond as 1.521 Å, bond angle N^i—C^i—C_α^i as 110.4° and dihedral angle N^i—C^i—C_α^i—C_β^i as 122.55°. We trained two models for different applications: Backbone only model (BBO) takes protein backbone conformation information as input which is suitable for full sequence prediction beginning with backbone structures only. Six input channels, which correspond to atoms C, C_α, N, O, OXT, and C_β have been used for this model. Backbone with sequence model (BBS) takes backbone information plus C_β atoms of nontarget residues labeled as one of the 20 amino acid types based on the sequence information. Twenty-six input channels, which correspond to atoms C, C_α, N, O, OXT, and C_β of target residue and 20 types of C_β (nontarget residue), have been used for BBS model. This model requires sequence information except target residue, which is suitable for predicting a single residue given the backbone structure and the amino acid types of the rest of the sequence.

The sequence recovery rates by our models trained on a dataset with 8120 protein chains are 42.20% for the BBO model (see Section 3 for details) and 47.63% for the BBS model. The recovery rates of the BBO and BBS models trained on another dataset with 17 040 protein chains are 46.50% and 52.15%, respectively.

2.1 | Datasets

We chose 10 149 protein structures (ID30) with the sequence's identity lower than 30% from PDB.[28] All the structures are determined by X-ray crystallography with the resolution better than 2.0 Å and do not have any DNA/RNA/UNK molecules. We randomly picked 90% protein chains (9135 protein chains) as training data (ID30TR), and the rest 10% (1014 protein chains) as test data (ID30TS). Another 50 protein chains (TS50) which are not included in the above dataset are used to test the models and compare with other methods. These 50 protein chains (pdb name + chain) are 1ahsA, 1bvyF, 1pdoA, 2va0A, 3ieyB, 2xr6A, 3ii2A, 1or4A, 2qdlA, 3nzmA, 3vjzA, 1eteA, 2a2lA, 2fvvA, 3l4rA, 1lpbA, 3nngA, 2cviA, 3gknA, 2j49A, 3fhkA, 3pivA, 3lqcA, 3gfsA, 3e8mA, 1dx5I, 3ny7A, 3k7pA, 2cayA, 1i8nA, 1v7mV, 1h4aX, 3t5gB, 3q4oA, 3a4rA, 2i39A, 3aqgA, 3ejfA, 3nbkA, 4gcnA, 2xdgA, 3gwiA, 3hklA, 3so6A, 3on9A, 4dkcA, 2gu3A, 2xcjA, 1y1lA, and 1mr1C. They were previously used by SPIN and Wang et al's model to compare with other methods.

We also generated another dataset ID90, which includes 21 071 protein structures from PDB with sequence identity lower than 90%. All the structures are determined by X-ray crystallography with the resolution better than 2.0 Å and do not have any DNA/RNA/UNK molecules. We used 17 044 protein structures as training data (ID90TR), and the rest 4027 protein structures as test data (ID90TS). The sequence identity of the structures between ID90TR and ID90TS are lower than 30% if the length difference ≤20%. The dataset TS50 is excluded from ID90TR. The BBO and BBS models trained on ID30TR are labeled as BBO_ID30 and BBS_ID30, and the ones trained on ID90TR are labeled as BBO_ID90 and BBS_ID90, respectively.

2.2 | Network architecture

As shown in Figure 1B, our CNN architecture has totally nine layers, which are described in detail as follows:

1. The input layer with dimension $18 \times 18 \times 18 \times N_{channel}$. $N_{channel}$ is the number of atom types.
2. Three parallel 3D convolution layers (conv3D) with rectified linear unit (ReLU) activation function. The first of those conv3D layers applies 16 channels of $2 \times 2 \times 2$ convolutional filters ($2 \times 2 \times 2 \times N_{channel} \times 16$) across the input with a stride as one to generate an output feature map with the same dimension as the input. This layer was designed to capture the information of covalent bonds, whose lengths range between 1.2 and 1.8 Å). The second conv3D layer applies 32 channels of $3 \times 3 \times 3$ convolutional filters ($3 \times 3 \times 3 \times N_{channel} \times 32$) with a stride as one, which was designed to recognize bond angles formed by two connected bonds. The last conv3D layer applies 32 channels of $4 \times 4 \times 4$ convolutional filters ($4 \times 4 \times 4 \times N_{channel} \times 32$) with a stride as one, which was aimed to capture information on dihedral angles.
3. All the outputs in layer 2 are combined ($18 \times 18 \times 18 \times 80$) and fed into a $2 \times 2 \times 2$ max pooling unit to get an output of dimension $9 \times 9 \times 9 \times 80$. Due to the max pooling process, the grid resolution changes to 2 Å.
4. A conv3D layer with 64 channels of $3 \times 3 \times 3$ filter is added to detect secondary structure motifs such as alpha helices and beta sheets.
5. A max pooling layer ($3 \times 3 \times 3$) is applied to get an output of ($3 \times 3 \times 3 \times 64$).
6. A flatten layer to convert the output of layer 5 to a simple vector.
7. A fully connected layer with ReLU activation function.
8. A dropout layer with a drop rate of 0.2.
9. A 20-dimensional SoftMax activation layer as the final output, which can be interpreted as the probability of 20 amino acid types of the target residue.

The neural network is constructed using the Keras library.[18] We used the rectified linear unit (ReLU) as the activation function

Model	BBO_ID30	BBO_ID90	BBS_ID30	BBS_ID90
Overall recovery rate	42.20%	46.50%	47.63%	52.15%
Fuzzy recovery rate	61.43%	64.90%	65.79%	69.53%
Recovery rate on TS50	38.71%	40.69%	43.20%	45.58%
Fuzzy recovery rate on TS50	58.81%	59.91%	62.15%	64.33%

TABLE 1 The average recovery rate (%) and fuzzy recovery rate (%) based on BLOSUM62 of different neural network models (BBO, BBS) trained with dataset ID30TR and ID90TR and test on dataset ID30TS (for ID30 models), ID90TS (for ID90 models), and TS50

Model	BBO_ID30	BBO_ID90	Wang's model	SPIN	SPIN2
Overall recovery rate	42.20%	46.50%	34.0%	30.7%	34.4%
Recovery rate on TS50	38.71%	40.69%	33.0%	30.3%	NA

Note: The overall recovery rates of our models, BBO_ID30 and BBO_ID90, tested on the dataset ID30TS (237 000 amino acid residues) and ID90TS (990 469 amino acid residues). The recovery rate of Wang et al's model is from fivefold cross-validation trained on a dataset consisting of 10 173 protein chains with sequence identity lower than 30%. The overall recovery rate of SPIN and SPIN2 is from 10-fold cross-validation trained on a dataset consisting of 1532 nonredundant proteins. These models are also tested on a small test dataset TS50 (50 protein chains) except SPIN2 which is tested on a dataset consisting of 500 protein chains including TS50.

TABLE 2 The average recovery rate of different models

for all layers except the output layer where we use SoftMax activation function. We used the categorical cross-entropy as the loss function and Adam for optimization. We trained our model for 7, 12, 7, 13 epochs for model BBO_ID30, BBO_ID90, BBS_ID30, and BBS_ID90, respectively, with a batch size of 200 and a learning rate of 0.01. We used batch normalization to reduce the internal covariate shift and dropout (drop rate = 0.2) to reduce overfitting in the neural network.

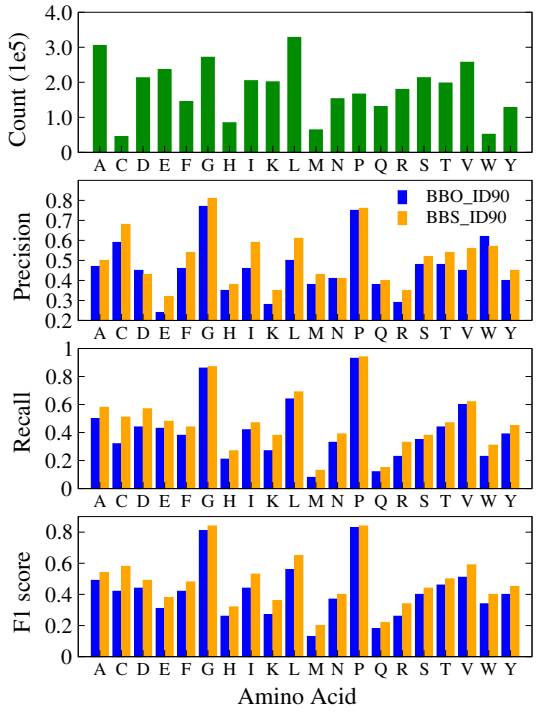

FIGURE 2 *Count* of each amino acids in ID90TR (top); *Precision*, *recall*, and *F1 score* of different amino acids of the networks BBO (blue) and BBS (orange) trained on ID90TR [Color figure can be viewed at wileyonlinelibrary.com]

3 | RESULTS

3.1 | Overall recovery rate

The average overall recovery rates of different models are shown in Table 1. The overall recovery rate is defined as the percent of amino acid residues that are predicted the same as the native sequence for a given protein structure. The BBS model trained on ID90TR has the best performance with 52.15% recovery rate on the test dataset ID90TS, and the BBS model trained on ID30TR has 47.63% recovery rate on ID30TS. The model trained on ID90TR increases the performance by more than 4%, indicating that the performance may be improved even further with more training data. Even though the sequence identity in ID90 data set is much higher than ID30, but we split the training data and test data to make sure that the sequence identity is lower than 30% between training and test dataset. So, the improvement comes mainly from the increased training data size (17 044 protein chains), which is nearly doubled compared to that of ID30TR (9135 protein chains).

We test all the models with the data set TS50 which is excluded from all the training data sets ID30TR and ID90TR. The recovery rates for BBO models are 40.69% and 38.71%, when trained on ID90TR and ID30TR, respectively. The performance of our method compares favorably to those of previous studies (Table 2). The average recovery rate on TS50 by Wang et al's model, trained on a dataset with sequence identity less than 30%, is 33.0%; The average recovery rates for SPIN and SPIN2 are 30.2% and 34.4%, respectively, on a 500 proteins dataset which contains TS50.

We also calculate a fuzzy recovery rate which is defined as the percent of amino acid residues that are predicted the same as the native residue types or as a substitutable residue type corresponding to BLOSUM62 (scores from the matrix are positive).[63] The fuzzy recovery rates for BBO models are 64.90% (BBO_ID90) and 61.43% (BBO_ID30), when trained on ID90TR and ID30TR, respectively, for BBS models are 69.53% (BBS_ID90) and 65.79% (BBS_ID30).

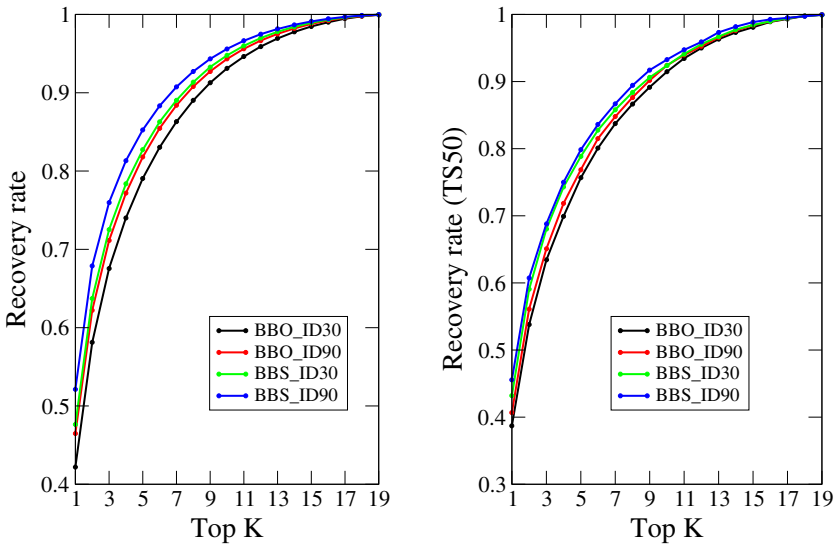

FIGURE 3 Top-K recovery rate of different models. Left: BBO_ID30 (black), BBO_ID90 (red), BBS_ID30 (green), BBS_ID90 (blue) models tested on ID30TS or ID90TS. Right: same models tested on TS50 [Color figure can be viewed at wileyonlinelibrary.com]

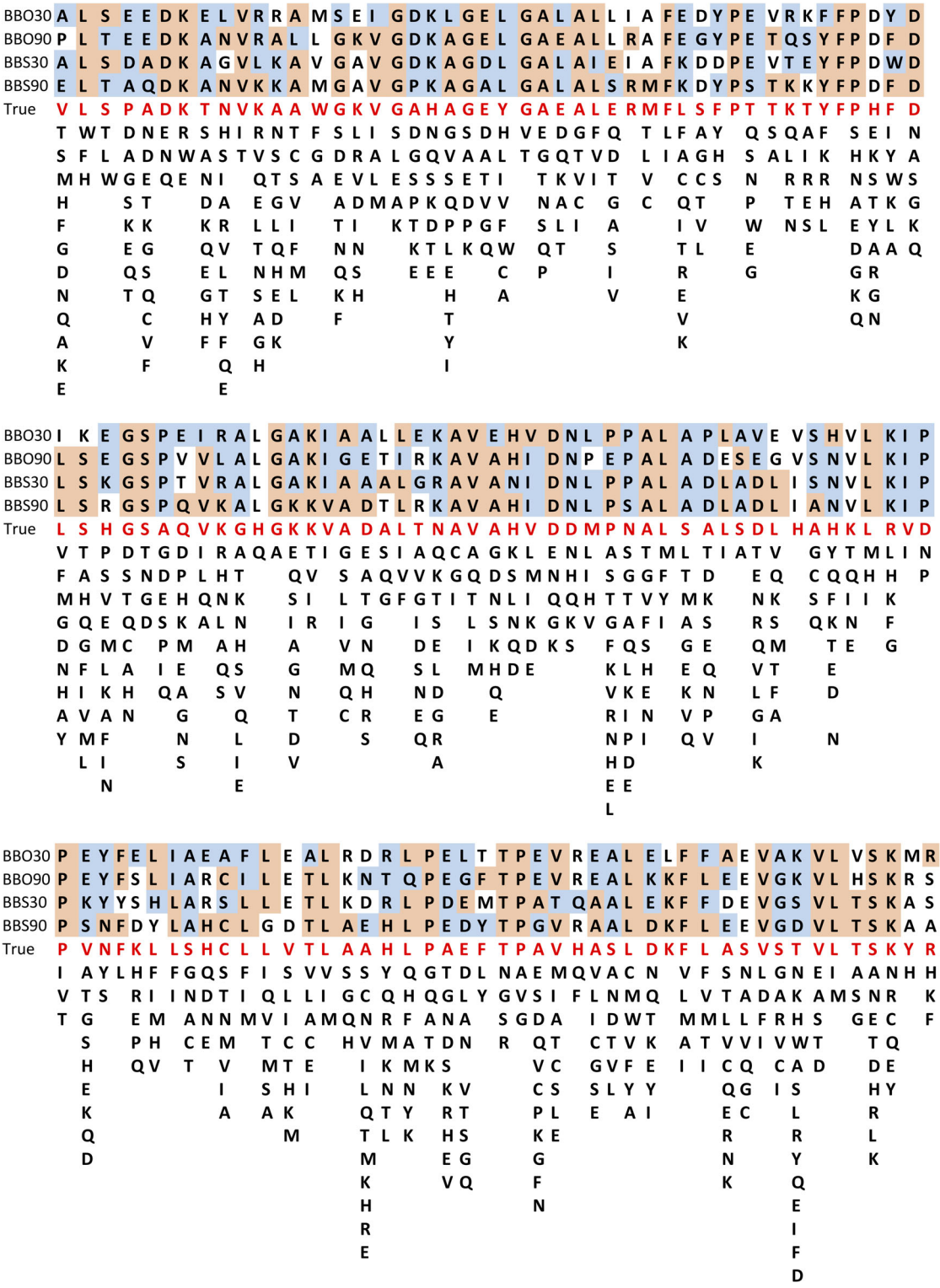

FIGURE 4 *Sequence prediction* by BBO_ID30 (BBO30), BBO_ID90 (BBO90), BBS_ID30 (BBS30), and BBS_ID90 (BBS90) model on human hemoglobin protein 1a3nA compared with true wild-type sequence (red). The amino acids shown below the true label row are alternative amino acids at the corresponding position based on the sequences which are similar (*P*-value = 0, twist = 0) with 1a3nA calculated by FATCAT.[65] An orange background indicates exactly correct prediction, and a blue background indicates that the prediction matches one of the alternative amino acids [Color figure can be viewed at wileyonlinelibrary.com]

3.2 | Accuracy on different amino acid types

Figures 2 and S1 show the distribution of each amino acid type in training data ID90TR and ID30TR, as well as the precision, recall, and F1 score of BBO and BBS models trained on these datasets. Precision is the percent of prediction that is correct, and recall is the percent of native residue that has been correctly predicted. There are large variations on the precisions and recalls among different amino acids types

with GLY and PRO having the best accuracy for both models, and GLN and MET having the worst accuracy.

3.3 | Top-K recovery rate

Since the last layer of our model outputs the probability of each amino acid, we could calculate the top-K recovery rate, where the prediction is considered as correct if the native residue type is within the top-K predictions based on the probabilities. As shown in Figure 3, the best performance is achieved by BBS_ID90 model, which gives the top-2, 3, 5, 10 recovery rates as 67.89%, 75.99%, 85.27%, and 95.60%, respectively. The BBO_ID30 yield top 2, 3, 5, 10 recovery rates as 58.12%, 67.57%, 79.07%, and 93.11%, respectively. The BBO_ID90 yields top 2, 3, 5, 10 recovery rates as 62.23%, 71.13%, 81.79%, and 94.32%, respectively. The fact that the top-10 predictions of our

TABLE 3 Case study

Model	BBO_ID30	BBO_ID90	BBS_ID30	BBS_ID90
1a3nA recovery rate	38.30%	45.39%	46.10%	58.16%
1a3nA soft recovery rate	81.56%	83.69%	82.98%	86.52%
1a3nB recovery rate	38.62%	51.72%	44.14%	62.76%
1a3nB soft recovery rate	78.62%	86.21%	80.00%	90.03%

Note: The recovery rate (%) and soft recovery rate (%) of the predicted sequences of deoxy human hemoglobin (pdb id: 1a3n, chain A and B) by different models.

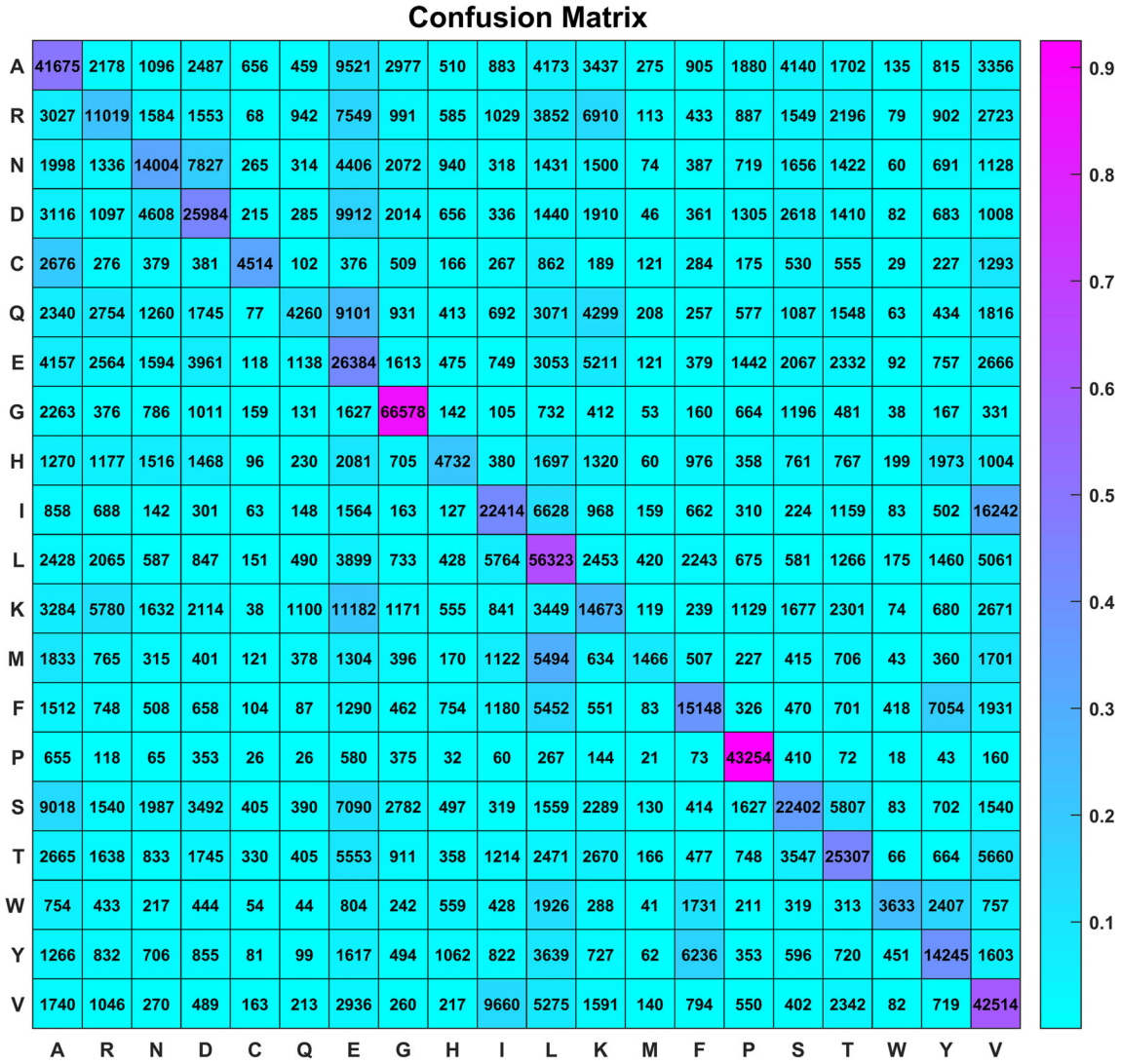

FIGURE 5 Confusion Matrix of BBO_ID90. The *y*-axis represents true labels and *x*-axis indicates predicted labels. The number in each entry indicates the number of times each amino acid was predicted as one of the 20 amino acids and the color shows the corresponding probability [Color figure can be viewed at wileyonlinelibrary.com]

models can reach well above 90% indicates that our models can separate very well those residues that may fit for a particular position in a structure and those residues that fit poorly for that position. We also calculate the top-K recovery rate of all the models on TS50 dataset, and the best BBO model (BBO_ID90) reaches 40.69%, 56.11%, 65.09%, 76.87%, and 92.41% for top-1, 2, 3, 5, 10 recovery rates, respectively. Comparing with SPIN and Wang et al's model, which give (30.2%, 45.3%, 55.2%, 67.7%, and 86.8%) and (33.0%, 48.7%, 58.5%, 71.7%, and 86.8%) for top-1, 2, 3, 5, 10 recovery rates, our model made substantial improvements than SPIN and Wang et al's model.

3.4 | Soft sequence recovery accommodating homologous sequences: a case study on deoxy human hemoglobin

It is well-known that proteins can undergo substantial mutations while maintaining their 3D structures and stabilities. If we make mutated proteins using predicted sequences not exactly the same as native sequences, the mutants may still fold to 3D structures similar to the input structure. In such cases, the predictions may still be considered as satisfactory. We used BBO_ID90 model to predict the protein sequences of deoxy human hemoglobin (pdb id: 1a3n, chain A and B), and obtained the recovery rate of 45.39% and 51.72% for 1a3nA and 1a3nB, respectively. Although it has been suggested that native sequences are optimal for their structures,[64] highly similar structures can still have significantly different sequences. We use FATCAT (Flexible structure AlignmenT by Chaining Aligned fragment pairs allowing Twists)[17] server to get all the sequences (77 protein chains for 1a3nA and 77 protein chains for 1a3nB) whose corresponding structures are similar to 1a3nA and 1a3nB using cutoff values of 0 for both *P*-value (very small *P*-values are rounded to zero) and twist. As shown in Figure 4 and Figure S2, the amino acid types that occur in at least one sequence (AA-homologous) is listed below the native sequence (red) of 1a3n for each position, and the predicted sequences by models BBO_ID30 (BBO30), BBO_ID90 (BBO90), BBS_ID30 (BBS30), and BBS_ID90 (BBS90) are shown above the native sequence. An orange background indicates exactly correct prediction, and a blue background indicates that the predictions fall into the AA-homologous set. We define the soft recovery rate as the proportion of predictions that fall into either native sequence or the AA-homologous set. The results are shown in Table 3, our BBO_ID90 model yields an 83.69% and 86.21% soft recovery rates for 1a3nA and 1a3nB, respectively, which are significantly higher than the sequence recovery rates (45.39% and 51.72%) without taking into account of homologous sequences.

3.5 | Confusion matrix

We plotted confusion matrix for BBO_ID90, BBO_ID30, BBS_ID30 and BBS_ID90 models in Figures 5, S3, S4, and S5, respectively. A cell (*x*, *y*) in the confusion matrix shows the number of times residue type *x* are predicted to be residue type *y*. Figure 6 shows a modified-confusion matrix for BBO_ID90 besides with BLOSUM62. In addition, the ones for BBO_ID30, BBS_ID30, and BBS_ID90 are shown in Figures S6-S8, respectively. The method of generating the modified-confusion matrices is given in Supporting Information. The two confusion matrices are highly similar with a *P*-value = 0 when tested using permutation test.

BLOSUM62

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	4	-1	-2	-2	0	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1	0	-3	-2	0
R	-1	5	0	-2	-3	1	0	-2	0	-3	-2	2	-1	-3	-2	-1	-1	-3	-2	-3
N	-2	0	6	1	-3	0	0	0	1	-3	-3	0	-2	-3	-2	1	0	-4	-2	-3
D	-2	-2	1	6	-3	0	2	-1	-1	-3	-4	-1	-3	-3	-1	0	-1	-4	-3	-3
C	0	-3	-3	-3	9	-3	-4	-3	-3	-1	-1	-3	-1	-2	-3	-1	-1	-2	-2	-1
Q	-1	1	0	0	-3	5	2	-2	0	-3	-2	1	0	-3	-1	0	-1	-2	-1	-2
E	-1	0	0	2	-4	2	5	-2	0	-3	-3	1	-2	-3	-1	0	-1	-3	-2	-2
G	0	-2	0	-1	-3	-2	-2	6	-2	-4	-4	-2	-3	-3	-2	0	-2	-2	-3	-3
H	-2	0	1	-1	-3	0	0	-2	8	-3	-3	-1	-2	-1	-2	-1	-2	-2	2	-3
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4	2	-3	1	0	-3	-2	-1	-3	-1	3
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4	-2	2	0	-3	-2	-1	-2	-1	1
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5	-1	-3	-1	0	-1	-3	-2	-2
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5	0	-2	-1	-1	-1	-1	1
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6	-4	-2	-2	1	3	-1
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7	-1	-1	-4	-3	-2
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4	1	-3	-2	-2
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5	-2	-2	0
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11	2	-3
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	-1
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

BBO_ID90

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	10	-3	-5	-3	1	-6	1	-5	-5	-9	-5	-2	-4	-6	-7	1	-5	-8	-6	-5
R	-3	9	-2	-5	-9	-1	2	-10	-2	-6	-2	5	-6	-7	-11	-4	-2	-8	-4	-4
N	-5	-2	12	5	-4	-5	0	-6	1	-14	-8	-2	-9	-8	-13	-2	-5	-10	-5	-10
D	-3	-5	5	12	-6	-7	3	-7	-2	-13	-9	-3	-12	-9	-8	-1	-5	-9	-6	-10
C	1	-9	-4	-6	18	-10	-8	-8	-5	-10	-7	-13	-4	-7	-13	-4	-4	-10	-8	-5
Q	-6	-1	-5	-7	-10	7	2	-12	-6	-11	-6	0	-5	-13	-15	-8	-6	-14	-11	-8
E	1	2	0	3	-8	2	11	-6	-2	-6	-3	5	-6	-7	-7	0	0	-7	-4	-3
G	-5	-10	-6	-7	-8	-12	-6	14	-10	-20	-13	-10	-13	-14	-11	-6	-11	-15	-13	-17
H	-5	-2	1	-2	-5	-6	-2	-10	13	-10	-5	-2	-8	0	-13	-5	-5	0	3	-7
I	-9	-6	-14	-13	-10	-11	-6	-20	-10	12	2	-7	-5	-5	-17	-15	-6	-9	-6	7
L	-5	-2	-8	-9	-7	-6	-3	-13	-5	2	11	-3	0	0	-13	-10	-6	-5	-2	-1
K	-2	5	-2	-3	-13	0	5	-10	-2	-7	-3	9	-8	-10	-11	-3	-1	-10	-6	-4
M	-4	-6	-9	-12	-4	-5	-6	-13	-8	-5	0	-8	8	-7	-15	-9	-7	-11	-9	-6
F	-6	-7	-8	-9	-7	-13	-7	-14	0	-5	0	-10	-7	13	-14	-10	-8	2	9	-5
P	-7	-11	-13	-8	-13	-15	-7	-11	-13	-17	-13	-11	-15	-14	17	-8	-14	-14	-15	-14
S	1	-4	-2	-1	-4	-8	0	-6	-5	-15	-10	-3	-9	-10	-8	10	1	-11	-7	-10
T	-5	-2	-5	-5	-4	-6	0	-11	-5	-6	-6	-1	-7	-8	-14	1	12	-11	-6	-1
W	-8	-8	-10	-9	-10	-14	-7	-15	0	-9	-5	-10	-11	2	-14	-11	-11	16	3	-9
Y	-6	-4	-5	-6	-8	-11	-4	-13	3	-6	-2	-6	-9	9	-15	-7	-6	3	14	-5
V	-5	-4	-10	-10	-5	-8	-3	-17	-7	7	-1	-4	-6	-5	-14	-10	-1	-9	-5	12

FIGURE 6 Modified confusion matrix for model BBO_ID90 (right) compared with BLOSUM62 (left). The two matrices are very similar (*P*-value = 0 from permutation test) [Color figure can be viewed at wileyonlinelibrary.com]

4 | CONCLUSIONS AND DISCUSSIONS

In this study, we developed a CNN model to address the inverse folding problem (IFP)—inferring the sequence of a protein given its 3D structure. Our method, called ProDCoNN (Protein Design with CNNs), achieved superior performance compared to those of earlier studies (Table 2).

Unlike the protein folding problem, the solution to IFP is not unique and the best solution is probably unknown. The native sequence is probably just one of the better solutions. Although it has been suggested that native sequences are likely optimal for a given structure,[65] the fact that many proteins can be engineered to more stable mutants[66-68] indicates that in terms of stability, many protein sequences are not optimal. The uncertainty on the optimal solution makes the evaluation of methods for IFP quite challenging. Given this uncertainty, we performed more in-depth analyses to gain more insight on the real performance and a better understanding of the strengths and weaknesses of our method as well.

The top-K recovery rates for BBO_ID90 model increase sharply from $K = 1$ to $K = 2$ and continues until $K = 5$, for which the recovery rate is above 80% (Figure 3). It is well-known that proteins can tolerate mutations on most of their sequence positions (see Figure 4 for an example). If the top-5 predictions correspond to those mutations that will not affect the protein stability significantly, then our effective sequence recover rate would be more than 80%. In most cases, residues similar in physical or chemical properties to the native ones are selected as the top-K predictions. This is further shown from the case study on deoxy human hemoglobin (Figure 4). The soft recovery rate, when homologous sequences were considered, can reach more than 80%. We hypothesized that a decent number of our predictions, although differing from the native sequences, may stabilize the given structure similarly or even more than the native sequences.

To further investigate whether some of our wrong predictions may actually be compatible with the native structures, we plotted confusion matrix for BBO_ID90 model in Figure 5. From this figure, we can see that the model made significant errors on distinguishing similar residues. For example, the cells corresponding to valine (VAL) and isoleucine (ILE) are the two residues with the highest number of errors, followed by GLU vs LYS and ALA vs SER. The errors made are not symmetric in that there are more ILE predicted as VAL than VAL predicted as ILE. Investigation of these errors may help us design better models to further improve the performance.

Interestingly, the confusion matrix (Figure 6 right) displays significant similarity (P-value = 0 using permutation test) with the well-known amino acid substation matrices (ie, BLOSUM62 matrix, Figure 6 left). This indicates that our models likely perform better than what they appeared using the stringent criterion of predicting the exact native amino acid types.

Our models have a relatively poor performance on residues with low abundance compared to other amino acid residue types, such as methionine (MET) and tryptophan (TRP). The proportion of MET is 1.7% in ID30TR. As the number of MET in training data ID90TR (the proportion is the same) increases, the precision and recall by BBO_ID90 model increased significantly (Figures 2 and 7). We believe that with more data available in the future, the performance of our models can still be improved by simply training with more data.

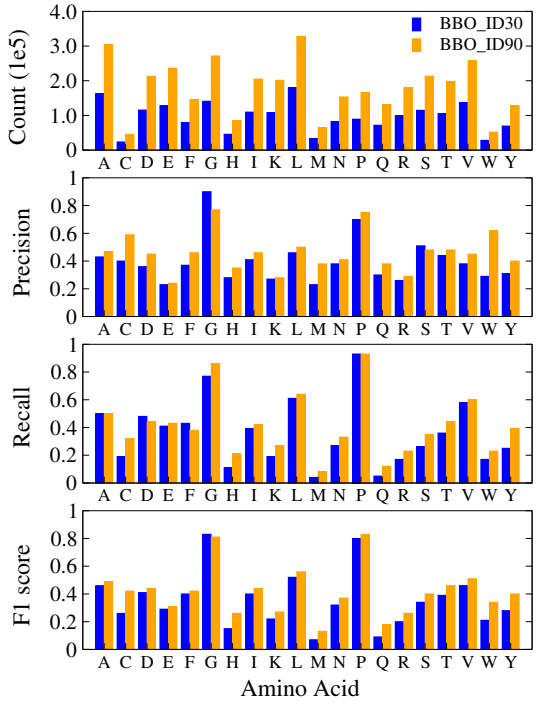

FIGURE 7 *Count* of each amino acids in ID30TR and ID90TR (top); *Precision*, *recall*, and *F1 score* of different amino acids of the networks BBO_ID30 (blue) and BBS_ID90 (orange) [Color figure can be viewed at wileyonlinelibrary.com]

ACKNOWLEDGMENTS

Research reported in this publication was supported by the National Institute of General Medical Sciences of the National Institute of Health under award number R01GM126558. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

ORCID

Jinfeng Zhang https://orcid.org/0000-0002-7429-7615

REFERENCES

1. Samish I. Achievements and challenges in computational protein design. *Methods Mol Biol*. 1529, 21-94, 2017. https://doi.org/10.1007/978-1-4939-6637-0_2
2. Korkegian A, Black ME, Baker D, Stoddard BL. Computational thermostabilization of an enzyme. *Science*. 308 (5723), 857-860, 2005. https://doi.org/10.1126/science.1107387.
3. Jiang L, Althoff EA, Clemente FR, et al. De novo computational design of retro-aldol enzymes. *Science*. 319 (5868), 1387-1391, 2008. https://doi.org/10.1126/science.1152692.
4. Röthlisberger D, Khersonsky O, Wollacott AM, et al. Kemp elimination catalysts by computational enzyme design. *Nature*. 2008;453:190-195. https://doi.org/10.1038/nature06879.
5. Murphy PM, Bolduc JM, Gallaher JL, Stoddard BL, Baker D. Alteration of enzyme specificity by computational loop remodeling and design. *Proc Natl Acad Sci*. 2009;106:9215-9220. https://doi.org/10.1073/pnas.0811070106.

6. Chen C-Y, Georgiev I, Anderson AC, Donald BR. Computational structure-based redesign of enzyme activity. *Proc Natl Acad Sci*. 2009; 106:3764-3769. https://doi.org/10.1073/pnas.0900266106.
7. Siegel JB, Zanghellini A, Lovick HM, et al. Computational design of an enzyme catalyst for a stereoselective bimolecular diels-alder reaction. *Science*. 329 (5989), 309-313, 2010. https://doi.org/10.1126/science.1190239.
8. Privett HK, Kiss G, Lee TM, et al. Iterative approach to computational enzyme design. *Proc Natl Acad Sci*. 2012;109:3790-3795. https://doi.org/10.1073/pnas.1118082108.
9. Slovic AM. Computational design of a water-soluble analog of phospholamban. *Protein Sci*. 2003;12:337-348. https://doi.org/10.1110/ps.0226603.
10. Yin H, Slusky JS, Berger BW, et al. Computational design of peptides that target transmembrane helices. *Science*. 315 (5820), 1817-1822, 2007. https://doi.org/10.1126/science.1136782.
11. Pabo C. Molecular technology: designing proteins and peptides. *Nature*. 1983;301:200. https://doi.org/10.1038/301200a0
12. Baker D. An exciting but challenging road ahead for computational enzyme design. *Protein Sci*. 2010;19:1817-1819. https://doi.org/10.1002/pro.481.
13. von der Osten C, Branner S, Hastrup S, et al. Protein engineering of subtilisins to improve stability in detergent formulations. *J Biotechnol*. 1993;28:55-68. https://doi.org/10.1016/0168-1656(93)90125-7.
14. Dahiyat BI, Sarisky CA, Mayo SL. De novo protein design: towards fully automated sequence selection. *J Mol Biol*. 1997;273:789-796. https://doi.org/10.1006/jmbi.1997.1341.
15. Desjarlais JR, Handel TM. De novo design of the hydrophobic cores of proteins. *Protein Sci*. 1995;4:2006-2018. https://doi.org/10.1002/pro.5560041006.
16. Raha K, Wollacott AM, Italia MJ, Desjarlais JR. Prediction of amino acid sequence from structure. *Protein Sci*. 9(6): 1106-1119, 2000. https://doi.org/10.1110/ps.9.6.1106.
17. Kuhlman B, Baker D. Native protein sequences are close to optimal for their structures. *Proc Natl Acad Sci*. 2000;12:10383-10388. https://doi.org/10.1073/pnas.97.19.10383
18. Dantas G, Kuhlman B, Callender D, Wong M, Baker D. A large scale test of computational protein design: folding and stability of nine completely redesigned globular proteins. *J Mol Biol*. 2003;332:449-460. https://doi.org/10.1016/S0022-2836(03)00888-X.
19. Hu C, Li X, Liang J. Developing optimal non-linear scoring function for protein design. *Bioinformatics*. 2004;20:3080-3098. https://doi.org/10.1093/bioinformatics/bth369.
20. Tsai H-HG, Tsai C-J, Ma B, Nussinov R. In silico protein design by combinatorial assembly of protein building blocks. *Protein Sci*. 2004; 13:2753-2765. https://doi.org/10.1110/ps.04774004.
21. Zhou H, Zhou Y. Fold recognition by combining sequence profiles derived from evolution and from depth-dependent structural alignment of fragments. *Proteins Struct Funct Genet*. 2005;58:321-328. https://doi.org/10.1002/prot.20308.
22. Li Q, Zhou C, Liu H. Fragment-based local statistical potentials derived by combining an alphabet of protein local structures with secondary structures and solvent accessibilities. *Proteins Struct Funct Bioinforma*. 2009;74:820-836. https://doi.org/10.1002/prot.22191.
23. Dai L, Yang Y, Kim HR, Zhou Y. Improving computational protein design by using structure-derived sequence profile. *Proteins Struct Funct Bioinforma*. 2010;78:2338-2348. https://doi.org/10.1002/prot.22746.
24. Li Z, Yang Y, Zhan J, Dai L, Zhou Y. Energy functions in de novo protein design: current challenges and future prospects. *Annu Rev Biophys*. 42, 315-335, 2013. https://doi.org/10.1109/AISMOT.2011.6159302.
25. Coluzza I. Computational protein design: a review. *J Phys Condens Matter*. 2017;29:143001. https://doi.org/10.1088/1361-648X/aa5c76.
26. von Heijne G. Protein evolution and design. *Annu Rev Biochem*. 2018;87:101-103. https://doi.org/10.1146/annurev-biochem-062917-012013.
27. Goodsell DS, Dutta S, Zardecki C, Voigt M, Berman HM, Burley SK. The RCSB PDB "molecule of the month": inspiring a molecular view of biology. *PLoS Biol*. 2015;13:e1002140. https://doi.org/10.1371/journal.pbio.1002140.
28. Rose PW, Prlić A, Altunkaya A, et al. The RCSB protein data bank: integrative view of protein, gene and 3D structural information. *Nucleic Acids Res*. 45 (D1), D271-D281, 2017. https://doi.org/10.1093/nar/gkw1000.
29. Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2012) ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems* 25, 1097-1105. https://doi.org/10.1145/3065386
30. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y. (2013) Over Feat: Integrated Recognition, Localization and Detection using Convolutional Networks. arXiv Prepr. arXiv: 1213.6229
31. Zeiler, M.D. and Fergus, R. (2014) Visualizing and understanding convolutional networks. *Lecture Notes in Computer Science*, 8689, 818-833. https://doi.org/10.1007/978-3-319-10590-1_53
32. Wu S, Zhong S, Liu Y. Deep residual learning for image steganalysis. *Multimed Tools Appl*. 2017;77:10437-10453. https://doi.org/10.1007/s11042-017-4440-4.
33. Graves, A., Fernández, S., Gomez, F. and Schmidhuber, J. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In ICML, 2006, 369-376. https://doi.org/10.1145/1143844.1143891
34. Hinton G, Deng L, Yu D, et al. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Process Mag*. 2012;29:82-97. https://doi.org/10.1109/MSP.2012.2205597.
35. Graves, A., Mohamed, A.R. and Hinton, G. Speech recognition with deep recurrent neural networks. ICASSP. IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, Canada; 6645-6649, 2013. https://doi.org/10.1109/ICASSP.2013.6638947
36. Abdel-Hamid O, Mohamed AR, Jiang H, Deng L, Penn G, Yu D. Convolutional neural networks for speech recognition. *IEEE Trans Audio, Speech Lang Process*. 2014; 10: 1533-1545. https://doi.org/10.1109/TASLP.2014.2339736.
37. Maas AL, Qi P, Xie Z, et al. Building DNN acoustic models for large vocabulary speech recognition. *Comput Speech Lang*. 2017;41:195-213. https://doi.org/10.1016/j.csl.2016.06.007.
38. Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit, Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, 55-60. https://doi.org/10.3115/v1/P14-5010
39. Goldberg Y. A primer on neural network models for natural language processing. *J Artif Intell Res*. 2016; 57:345–420.
40. Young T, Hazarika D, Poria S, Cambria E. Recent trends in deep learning based natural language processing. *IEEE Comput Intell Mag*. 2018; 13:55-75. https://doi.org/10.1109/MCI.2018.2840738.
41. Devlin, Jacob and Chang, Ming-Wei and Lee, Kenton and Toutanova, K. (2018) BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Volume 1, 2019; 4171-4186. https://doi.org/10.18653/v1/N19-1423
42. Kulkarni, A. and Shivananda, A. Deep learning for NLP. *Natural Language Processing Recipes*. Apress, 2019, 185-227. https://doi.org/10.1007/978-1-4842-4267-4_6
43. LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*. 2015;521:436-444. https://doi.org/10.1038/nature14539.
44. Schmidhuber J. Deep learning in neural networks: an overview. *Neural Netw*. 2015;61:85-117. https://doi.org/10.1016/j.neunet.2014.09.003.

45. Faraggi E, Zhang T, Yang Y, Kurgan L, Zhou Y. SPINE X: improving protein secondary structure prediction by multistep learning coupled with prediction of solvent accessible surface area and backbone torsion angles. *J Comput Chem*. 2012;33:259-267. https://doi.org/10.1002/jcc.21968.
46. Sønderby, S.K. and Winther, O. (2014) Protein secondary structure prediction with long short term memory networks. *arXiv Prepr*. arXiv:1412.7828.
47. Li, Z. and Yu, Y. Protein secondary structure prediction using cascaded convolutional and recurrent neural networks. IJCAI International Joint Conference on Artificial Intelligence, New York, 2016, 2560-2567. arXiv Prepr.arXiv:1604.07176.
48. Busia, A. and Jaitly, N. (2017) Next-step conditioned deep convolutional neural networks improve protein secondary structure prediction. *arXiv Prepr*. arXiv:1702.03865v1.
49. Fang C, Shang Y, Xu D. MUFOLD-SS: new deep inception-inside-inception networks for protein secondary structure prediction. *Proteins Struct Funct Bioinforma*. 2018;86:592-598. https://doi.org/10.1002/prot.25487.
50. Tegge AN, Wang Z, Eickholt J, Cheng J. NNcon: improved protein contact map prediction using 2D-recursive neural networks. *Nucleic Acids Res*. 2009;37:W515-W518. https://doi.org/10.1093/nar/gkp305.
51. Fariselli P, Olmea O, Valencia A, Casadio R. Prediction of contact maps with neural networks and correlated mutations. *Protein Eng des Sel*. 2001;14:835-843. https://doi.org/10.1093/protein/14.11.835.
52. Di lena P, Nagata K, Baldi P. Deep architectures for protein contact map prediction. *Bioinformatics*. 2012;28:2449-2457. https://doi.org/10.1093/bioinformatics/bts475.
53. Wang S, Sun S, Li Z, Zhang R, Xu J. Accurate de novo prediction of protein contact map by ultra-deep learning model. *PLoS Comput Biol*. 2017;13:e1005324. https://doi.org/10.1371/journal.pcbi.1005324.
54. Tian K, Shao M, Wang Y, Guan J, Zhou S. Boosting compound-protein interaction prediction by deep learning. *Methods*. 2016;110:64-72. https://doi.org/10.1016/j.ymeth.2016.06.024.
55. Hua L, Quan C. A shortest dependency path based convolutional neural network for protein-protein relation extraction. *Biomed Res Int*. 2016;2016:1-9. https://doi.org/10.1155/2016/8479587.
56. Ragoza M, Hochuli J, Idrobo E, Sunseri J, Koes DR. Protein-ligand scoring with convolutional neural networks. *J Chem Inf Model*. 2017;57:942-957. https://doi.org/10.1021/acs.jcim.6b00740.
57. Sun T, Zhou B, Lai L, Pei J. Sequence-based prediction of protein protein interaction using a deep-learning algorithm. *BMC Bioinform*. 2017;18:277. https://doi.org/10.1186/s12859-017-1700-2.
58. Li Z, Yang Y, Faraggi E, Zhan J, Zhou Y. Direct prediction of profiles of sequences compatible with a protein structure by neural networks with fragment-based local and energy-based nonlocal profiles. *Proteins Struct Funct Bioinforma*. 2014;82:2565-2573. https://doi.org/10.1002/prot.24620.
59. O'Connell J, Li Z, Hanson J, et al. SPIN2: predicting sequence profiles from protein structures using deep neural networks. *Proteins Struct Funct Bioinforma*. 2018;86:629-633. https://doi.org/10.1002/prot.25489.
60. Wang J, Cao H, Zhang JZH, Qi Y. Computational protein design with deep learning neural networks. *Sci Rep*. 2018;8:6349. https://doi.org/10.1038/s41598-018-24760-x
61. LeCun Y, Boser B, Denker JS, et al. Handwritten digit recognition with a Back-propagation network. *Integr Vlsi J*. 2000;39:149. https://doi.org/10.1111/dsu.12130.
62. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. *Proc IEEE*. 1998;86:2278-2324. https://doi.org/10.1109/5.726791.
63. Henikoff S, Henikoff JG. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A*. 1992;89:10915-10919. https://doi.org/10.1073/pnas.89.22.10915
64. Chollet, F. (2015) Keras. https://keras.io.
65. Ye Y, Godzik A. Flexible structure alignment by chaining aligned fragment pairs allowing twists. *Bioinformatics*. 2003;19:ii246-ii255. https://doi.org/10.1093/bioinformatics/btg1086
66. Liu H-L, Wang W-C. Protein engineering to improve the thermostability of glucoamylase from aspergillus awamori based on molecular dynamics simulations. *Protein Eng des Sel*. 2003;16:19-25. https://doi.org/10.1093/proeng/gzg007.
67. Culyba EK, Price JL, Hanson SR, et al. Protein native-state stabilization by placing aromatic side chains in N-glycosylated reverse turns. *Science*. 2011; 331 (6017), 571-575. https://doi.org/10.1126/science.1198461.
68. Fersht AR, Serrano L. Principles of protein stability derived from protein engineering experiments. *Curr Opin Struct Biol*. 1993;3:75-83. https://doi.org/10.1016/0959-440X(93)90205-Y.

SUPPORTING INFORMATION

Additional supporting information may be found online in the Supporting Information section at the end of this article.

How to cite this article: Zhang Y, Chen Y, Wang C, et al. ProDCoNN: Protein design using a convolutional neural network. *Proteins*. 2020;1–11. https://doi.org/10.1002/prot.25868