Received: 12 September 2018 | Revised: 17 June 2019 | Accepted: 6 July 2019

DOI: 10.1002/prot.25780

PROTEINS WILEY

RESEARCH ARTICLE

A deep dense inception network for protein beta-turn prediction

Chao Fang[1] | Yi Shang[1] | Dong Xu[1,2]

[1]Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, Missouri

[2]Christopher S. Bond Life Sciences Center, University of Missouri, Columbia, Missouri

Correspondence
Yi Shang and Dong Xu, Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO.
Emails: shangy@missouri.edu (Y.S.) and xudong@missouri.edu (D.X.)

Funding information
National Institute of General Medical Sciences, Grant/Award Number: GM100701

Abstract

Beta-turn prediction is useful in protein function studies and experimental design. Although recent approaches using machine-learning techniques such as support vector machine (SVM), neural networks, and K nearest neighbor have achieved good results for beta-turn prediction, there is still significant room for improvement. As previous predictors utilized features in a sliding window of 4-20 residues to capture interactions among sequentially neighboring residues, such feature engineering may result in incomplete or biased features and neglect interactions among long-range residues. Deep neural networks provide a new opportunity to address these issues. Here, we proposed a deep dense inception network (DeepDIN) for beta-turn prediction, which takes advantage of the state-of-the-art deep neural network design of dense networks and inception networks. A test on a recent BT6376 benchmark data set shows that DeepDIN outperformed the previous best tool BetaTPred3 significantly in both the overall prediction accuracy and the nine-type beta-turn classification accuracy. A tool, called MUFold-BetaTurn, was developed, which is the first beta-turn prediction tool utilizing deep neural networks. The tool can be downloaded at http://dslsrv8.cs.missouri.edu/~cf797/MUFoldBetaTurn/download.html.

KEYWORDS

deep learning, deep neural network, dense network, inception network, protein beta turn, protein structure prediction

1 | INTRODUCTION

Protein tertiary structure prediction is an important and challenging problem, which has been an active research topic in the past 50 years.[1-3] As it is challenging to predict the protein tertiary structure directly from a protein primary sequence, this problem has been divided into small subproblems, such as protein secondary structure prediction. The protein secondary structures are divided into three classes: alpha-helix, beta-sheets, and coil.[4] The coil region can be classified as tight turns, bulges, and random coil structures.[5] The tight turns are further classified into alpha-turns, beta-turns, gamma-turns, delta-turns, and pi-turns.[6] Among these tight turns, beta-turns represent the most abundant type in proteins. For example, in the BT6376[7] data set, we found 126 016 beta-turns (9%) out of 1 397 857 amino acids. By definition, a beta-turn contains four consecutive residues (denoted by i, $i + 1$, $i + 2$, and $i + 3$) if the distance between the $C\alpha$ atom of residue i and the $C\alpha$ atom of residue $i + 3$ is less than 7 Å and if the central two residues are not helical.[8] An alternative but more precise definition of beta-turn is the possession of an intra-main-chain hydrogen bond between the CO of residue i and the NH of residue $i + 3$[9] (see Figure 1 for an illustration). There are nine types of beta-turns, which are classified based on the dihedral angles of two central residues in a turn[10] as shown in Table 1. Beta-turns can be assigned from a Protein Data Bank (PDB) structure by using the PROMOTIF software.[10] Beta-turns play an important role in mediating interactions between peptide ligands and their receptors.[11] In protein design, loop segments and

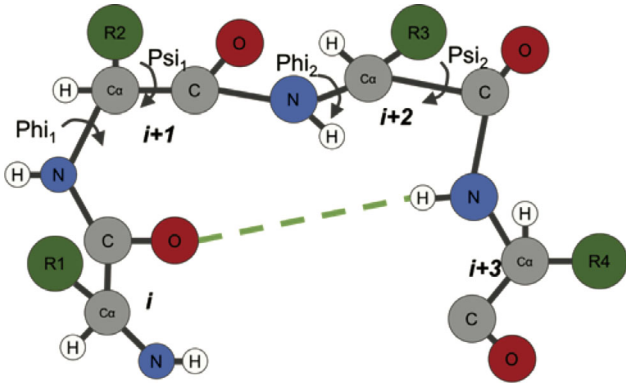

FIGURE 1 An illustration of what a beta-turn is. C, O, N, and H represent carbon, oxygen, nitrogen, and hydrogen atoms, respectively. R represents a side chain. A dashed line represents a hydrogen bond

TABLE 1 Nine types of beta-turns and their dihedral angles of central residues in degrees

Turn type	Phi_1	Psi_1	Phi_2	Psi_2
I	−60	−30	−90	0
I	60	30	90	0
II	−60	120	80	0
II′	60	−120	−80	0
IV	−61	10	−53	17
VIII	−60	−30	−120	120
VIb	−135	135	−75	160
VIa1	−60	120	−90	0
VIa2	−120	120	−60	0

Note: The locations of Phi_1, Psi_1, Phi_2, and Psi_2 are illustrated in Figure 1.

hairpins can be formed by introducing beta-turns in proteins and peptides.[12] Hence, it is important to predict beta-turns from a protein sequence.[13]

The early predictors[10,14,15] used statistical information derived from protein tertiary structures to predict beta-turns based on the positional frequencies of amino acid residues. Zhang and Chou[16] further observed the pairing between the first and fourth residues and between the second and the third residues, which plays an important role in beta-turn formation. They proposed the 1-4 and 2-3 correlation model to predict beta-turns.[16] Later, Chou[17] applied a sequence-coupled approach based on the first-order Markov chain to further improve their prediction model. Kaur and Raghava[13] developed a web server, called BetaTPred, which implemented this model and achieved an Matthew correlation coefficient (MCC) of 0.26 in beta-turn prediction.

McGregor et al[18] used neural networks to predict beta-turns, which is the first machine-learning approach for beta-turn prediction, and they achieved an MCC of 0.20. Shepherd et al[19] developed BTPred using secondary structure information as input and achieved an MCC of 0.34. Kim[20] applied a K-nearest neighbor method for beta-turn prediction and improved MCC to 0.40. Fuchs and Alix[21] further improved the MCC to 0.42 by incorporating multiple features such as propensities, secondary structures, and position-specific scoring matrix (PSSM). Kaur and Raghava[22] developed the BetaTPred2 server, which used a two-layer neural network with an MCC of up to 0.43. Kirschner and Frishman[23] developed MOLEBRNN using a novel bidirectional recurrent neural network, with an MCC of up to 0.45. Hu and Li[24] used a support vector machine and incorporated features such as increment of diversity, position conservation scoring function, and secondary structure to raise the MCC up to 0.47. Zheng and Kurgan[25] used the predicted secondary structures from PSIPRED,[26] JNET,[27] TRANSEEC,[28] and PROTEUS2[29] to improve the performance. Kountouris and Hirst[30] used predicted dihedral angles along with PSSM and predicted secondary structures to achieve an MCC of 0.49. Petersen et al[31] developed the NetTurnP server with an MCC of 0.50 by using independent four models for predicting four positions in a beta-turn. Singh et al[7] developed the BetaTPred3 server to achieve an MCC of 0.51 using a random forest method, which was the most accurate method before our work.

The above-mentioned machine-learning methods achieved some successful results in beta-turn prediction. However, there is significant room for improvement, particularly in predicting nine types of beta-turns. Most of these previous methods relied on a sliding window of 4-10 amino acid residues to capture short interactions. Also, previous neural networks with one or two layers (shallow neural networks) could not extract high-level features from input data sets. So far, no deep neural networks have been applied to beta-turn prediction. Deep neural networks can learn representations of data with multiple levels of abstraction,[32] which provides a new opportunity to this old research problem.

Our previous studies[33-35] have successfully applied the stacked convolutional neural network (CNN), the inception module,[36] and the residual module[37] to protein sequence analysis and prediction problems. Following our previous work, here we propose a new deep neural network architecture called deep dense inception network (DeepDIN) for beta-turn prediction. The contributions are presented as follows: (a) MUFold-BetaTurn is the first beta-turn prediction software to utilize the deep-learning framework and outperformed the previous best predictor BetaTPred3[7]; (b) we employed strategies such as balanced learning and transfer learning to tackle the problem of small sizes and imbalanced data sets for deep learning, which may provide a good example for some other deep-learning applications in bioinformatics; and (c) we provide a free standalone software for the research community to use.

2 | MATERIALS AND METHODS

2.1 | Preliminaries and problem formulation

To make an accurate prediction, it is important to provide useful input features to machine-learning models. In our method, we carefully designed feature matrices corresponding to the primary amino acid sequence of a protein. Specifically, our feature sets include: (a) a physicochemical feature set describing properties of amino acids, (b) a hidden Markov models (HMMs): HMM-HMM- based lightning-fast

iterative sequence search (HHBlits)[38] profile, (c) prediction of eight state secondary structures from MUFold-SS,[33] and (d) a shape string predicted by Frag1D.[39]

Physicochemical features describe hydrophobic, steric, and electric properties of amino acids and provide useful information for protein sequence analysis and prediction. By using physicochemical properties, protein sequences are represented as an informative dense matrix. The physicochemical feature matrix consists of seven physicochemical properties as defined by Heffernan et al,[40] plus a number 0 or 1 representing the existence of an amino acid at this position as an input (called *NoSeq* label). The reason for adding the *NoSeq* label is because the proposed deep neural networks are designed to take a fixed-size input, such as a sequence length of 700 residues in our experiment. To run a protein sequence shorter than 700 through the network, the protein sequence will be padded at the end with 0 values and the *NoSeq* label is set to 1. If the protein is longer than 700 residues, it can be split into multiple segments, each shorter than 700 residues. Hence, a protein sequence will be represented as a 700-by-8 matrix, which is the first input feature set for DeepDIN.

The second set of useful features comes from the protein profiles generated using HHBlits.[38] In our experiments, the HHBlits software used the database uniprot20_2013_03, which can be downloaded from http://wwwuser.gwdg.de/~compbiol/data/hhsuite/databases/hhsuite_dbs/. It is known that a larger and most recent sequence library will improve the prediction results. The reason why we used an old version of the database is to have an unbiased comparison with other predictors. Larger sequence databases with recent sequences will be updated and applied to our software tool. The profile values were transformed by the sigmoid function into the range (0, 1). Each amino acid in the protein sequence is represented as a vector of 31 real numbers, of which 30 are from amino acids HHBlits profile values and one is a *NoSeq* label in the last column. The HHBlits profile contains amino acids and some transition probabilities: "A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y, M->M, M->I, M->D, I->M, I->I, D->M, D->D, Neff, Neff_I, and Neff_D." HHBlits' profiles are more sensitive than PSI-BLAST profiles and provide useful evolutionary features for the protein sequence. A HHBlits profile is represented as a 700-by-30 matrix, which is the second input feature for DeepDIN.

The third set of features, the predicted shape strings, comes from Frag1D.[39] For each protein sequence, the Frag1D predicts the classical three-state secondary structure, and three- and eight-state shape string. A classical three-state secondary structure contains an H (helix), S (sheet), and R (random loop). Eight-state shape string labels are defined as: R (polyproline type alpha structure), S (beta sheet), U, V (bridging regions), A (alpha helices), K (3_{10} helices), G (almost entirely glycine), and T (turns). Shape strings[41] describe a one-dimensional (1D) string of symbols representing the protein backbone Psi-Phi torsion angles. They include regular secondary structure elements, where shape "A" corresponds to alpha helix and shape "S" corresponds to beta strands. In addition, shape strings classify the random loop regions into several states that contain much more conformation information. For the Frag1D predicted result, each amino acid in the protein sequence is represented as a vector of 15 numbers: three are from the classical three-state secondary structures, three are from the three-state shape strings, eight are from the eight-state shape strings, and one *NoSeq* label is in the last column. The predicted classical three-state secondary structure feature set is represented in one-hot encoding as follows—helix: (1,0,0), strand: (0,1,0), and loop: (0,0,1). The same encoding applies to three- and eight-state shape string features. Hence, a Frag1D result is represented as a 700-by-15 matrix, which is the third input feature for DeepDIN.

The fourth set of features comes from our secondary structure prediction tool: MUFold-SS.[33] MUFold-SS achieved state-of-the-art performance in an eight-state secondary structure prediction, and it should provide useful features in beta-turn prediction tasks. The eight-state secondary structures have the following components: H (alpha helix), B (beta bridge), E (extended strand), G (3-helix), I (5 helix), T (hydrogen bonded turn), and S (bend). Hence, an eight-state predicted secondary structures will be represented in one-hot encoding as a 700-by-8 matrix, which is the fourth input feature for DeepDIN.

Protein beta-turn prediction is a classification problem. To be specific, it can be formulated as a residue-level prediction or turn-level prediction as first proposed by.[7]

- Residue-level prediction: To predict whether each amino acid has a class label of a turn or nonturn. Here, the predicted output is a 700-by-3 matrix, where "700" is the sequence length 700 and "3" for two-state labels plus 1 *NoSeq* label.
- Turn-level prediction: At the turn-level, a sliding window of four residues was used to generate the turn-level data sets. And the overall predicted beta-turn output of a protein sequence is represented as a fixed-size matrix, with a (700-4+1) by 3 dimension, for two-state labels plus 1 *NoSeq* label. For a nine-class classification problem, the label is turn of a specific type or nonturn.

The evaluation metric for beta-turn prediction typically is MCC, instead of accuracy, as the accuracy only considers the true positive and false positive without the true negative and false negative, and non-beta-turns (negative data) dominate the data. MCC can be calculated as follows:

$$\mathrm{MCC} = \frac{\mathrm{TP}*\mathrm{TN} - \mathrm{FP}*\mathrm{FN}}{\sqrt{(\mathrm{TP}+\mathrm{FP})(\mathrm{TP}+\mathrm{FN})(\mathrm{TN}+\mathrm{FP})(\mathrm{TN}+\mathrm{FN})}} \quad (1)$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

2.2 | New DeepDIN for protein beta-turn prediction

In this section, a new DeepDIN architecture is presented. The architecture makes use of deep inception[36] networks and dense networks.[42] Both network designs have shown promising performance in image recognition tasks.[36,42] Deep inception networks[36] consist of inception modules, which contain convolutional filters of different kernel sizes to capture details from the input (such as images, voice

recordings, or in this case protein sequence features) at varied scales, (5 × 5, 3 × 3, 1 × 1, etc.). The advantages of deep inception networks are to explore features at multiple scales together and discover high-level features. Deep dense network[42] connects each layer to every other layer in a feed-forward fashion. For each layer, the feature-maps of all previous layers are used as inputs. Densely connected networks have several advantages over traditional deep convolutional networks; in particular, it can alleviate the vanishing-gradient problem, strengthen feature propagation, and accommodate feature reuse. In this article, we proposed a new network called DeepDIN, which takes advantage of both network designs, that is, deep inception networks and deep dense networks as shown in Figure 2. The overall beta-turn prediction pipeline contains the following steps:

1. Given a protein sequence, generate four sets of features: physicochemical feature, profile features from HHBlits, predicted shape string (using Frag1D), and predicted eight-state secondary structure (using MUFold-SS).
2. Perform the convolution operation on each feature to get the convolved feature map.
3. Concatenate four convolved feature maps along the feature dimension.
4. Feed the concatenated feature map into the stringed dense inception blocks. In between, there is a convolutional layer acting as the transition layer.
5. Predict beta-turn (either turn or nonturn) in the last dense layer, which uses Softmax to normalize the output.

Figure 3 shows the details of a dense inception block of DeepDIN. It consists of four small inception blocks, and each inception block is fully connected to the latter inception blocks. In other words, a dense inception module is constituted by connecting each inception layer to every other inception layer after it in a feed-forward fashion. The design of dense inception modules can extract nonlocal interactions of residues over a diverse range in a more effective way. Adding more dense inception blocks is possible but requires more computer memory when running the model.

Each convolution layer, such as "Conv (3)" in Figure 3, consists of four operations sequentially: (a) an 1D-convolution operation using kernel size three; (b) the batch normalization technique[43] for speeding up the training process and acting as a regularizer; (c) the activation operation, rectified linear unit[44]; and (d) the dropout operation[45] to prevent the neural network from overfitting by randomly dropping neurons during the deep network training process so that the network can avoid or reduce coadapting. DeepDIN was implemented, trained, and tested using TensorFlow and Keras. All experiments were performed on an Alienware Area-51 desktop computer equipped with Nvidia Titan-X GPU (11 GB memory). In our experiments, many network hyperparameters and training parameters were tried. For the results reported, the dropout rate was set at 0.4. The optimizer used during training is Adam,[46] which can control the learning rate dynamically for network weight updates. There are nine beta-turn classifications, and the observations for a certain class can be as many as 40 000 or as little as 100. In different classification tasks, the batch size varies from 50 to 200. The maximum number of epochs is set up to 100. The training time varies from 2 to 5 hours depending on the data size when training different classification models.

2.3 | Apply transfer learning and balanced learning to DeepDIN to handle imbalanced small data set

Deep learning usually requires a large amount of data to train the networks well. Here, as some of the beta-turn classes have only a small

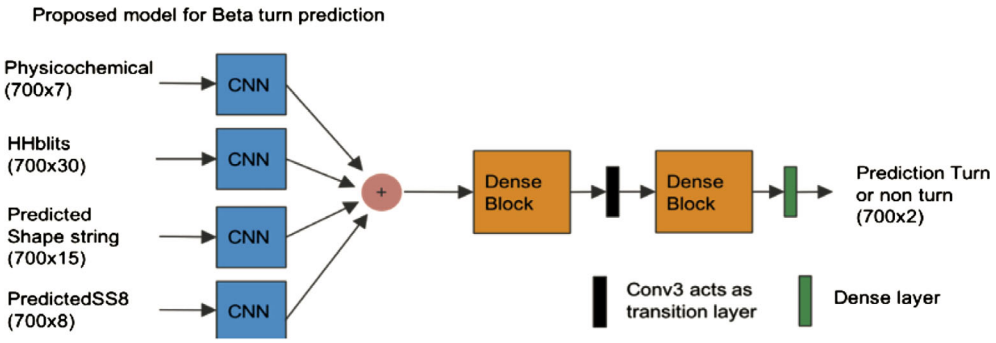

FIGURE 2 Overall DeepDIN network architecture for beta-turn prediction [Color figure can be viewed at wileyonlinelibrary.com]

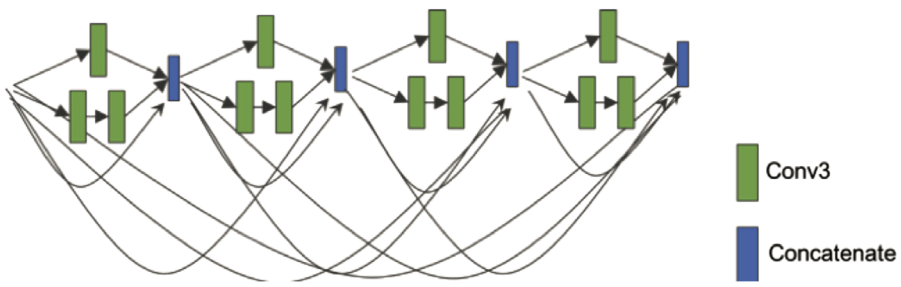

FIGURE 3 An illustration of a dense inception module in DeepDIN. Each dense inception module contains four basic inception modules, each of which is fully connected to every other inception layer after it in a feed forward fashion [Color figure can be viewed at wileyonlinelibrary.com]

amount of data, the following deep-learning strategy was used to address the small data sets classification problem.

2.3.1 | Balanced learning

Many researchers have studied the problem of imbalanced data learning and proposed ways of balancing the data set by using either oversampling like SMOTE[47] or undersampling like Tomek Links.[48] The beta-turn data are highly unbalanced with a very small fraction of positive examples, which would cause a deep neural network like DeepDIN prone to predict everything belonging to the negative class. In this experiment, the protein beta-turn data classified as positive are not large; thus, any up-sampling or down-sampling may cause loss or pollution of useful information. Rather than resampling the training examples, a balanced learning approach was adapted to handle the problem. The weighted cross entropy was used as the loss function to address the issue caused by the small sample size by rescaling the prediction of each class by its weight. In this implementation, class weights were calculated using the training data and assigned using the Scikit-learn[49] toolbox. The balanced class weights are given by $n\_samples/(n\_classes * bincount(y))$, where y is an array of original class labels per sample, and "bincount" is a built-in function from the Scikit-learn toolbox to calculate the number of bins.

2.3.2 | Transfer learning

We applied transfer learning to handle the limited number of beta-turn data used in the training set. The idea of transfer learning originally came from the image classification problem.[50] This technique was proposed to handle the insufficient size of a data set that can be used to train a deep neural network. In our study, as there are nine classes of beta-turns, and especially as those in VIa1, VIa2, and VIb contain only a few hundred data points, the amount of data belonging to each class may not produce a model with the ability to extract features or to be well generalized. To solve this problem, the DeepDIN model trained for classifying two-class beta-turns was used as the pretrained model when separate DeepDIN models were trained for nine-class classifications. The pretrained weights were loaded into a separate nine-class classification model as initial weights. Notably, the pretrained model here is the beta-turn model used to classify the two-class beta-turn problem. As that model has "observed" some generic features of what a beta-turn is, it should be useful to many specific nine-class beta-turn classification tasks. Then, to train each individual class model, each individual class training set was used to further train the pretrained network. It is possible to fix the earlier layers of the deep networks due to the overfitting issue and only fine-tune the latter layers of the network. The samples used to pretrain the network did not overlap with the testing samples. Without the pretraining process, the model training might not converge, especially for those classes with limited training samples. After many trials, the final network was fine-tuned without freezing the weights from lower level of the network.

For the learning rates during the transfer learning, a smaller learning rate (0.005) and batch size (10) were used to train the network and fine-tune the network weights. The reason is that the pretrained network weights are relatively good, a smaller learning rate will not distort them too much before it converges.

TABLE 2 Summary of beta-turn benchmark data sets BT426 and BT6376

	BT426	BT6376
Number of protein chains	426	6376
Sequence identity cutoff	25%	30%
X-ray resolution	2 Å	2 Å
Two-class prediction	Residue level; turn level	Residue level; turn level
Nine-class prediction		Residue level; turn level

2.4 | Benchmark data sets

The following two publicly available benchmark data sets were used in our experiments. Previous predictors reported the 5-fold cross-validation performance on an earlier benchmark BT426. The most recent benchmark BT6376 was created by Singh et al,[7] and they reported their webserver BetaTPred3's[7] 5-fold cross-validation performance on benchmark BT6376. To compare with previous predictors, we also performed the same experiment of 5-fold cross-validation on both data sets. We summarize the benchmark data sets in Table 2.

1. BT426[51] is a data set commonly used for benchmarking beta-turn prediction methods. BT426 contains 426 protein chains with 25% sequence identity cutoffs, and X-ray structures of a resolution better than 2.0 Å. This benchmark was used to compare the performance of several previous predictors, such as BetaTPred3[7] and NetTurnP.[31] In this work, 5-fold cross-validation experiments on BT426 were performed and results were compared against other predictors.
2. BT6376[7] is a public benchmark containing 6376 nonhomologous protein chains. No two protein chains have more than 30% sequence identity. The structures of these proteins were determined by X-ray crystallography at 2.0 Å resolution or better. Each chain contains at least one beta-turn. The beta-turn labels were assigned by the PROMOTIF program.[10] This benchmark provides data sets for both the two-class beta-turn classification and the nine-class beta-turn classification. For the nine-class beta-turn classification, labels were annotated by using PROMOTIF. Table 3 shows a list of class sizes in BT6376, which indicates that the data set is very imbalanced, as the turns samples are only 0.3% to 3.1% of the non-turns samples.

3 | RESULTS AND DISCUSSION

In this section, extensive experimental results of the proposed deep neural networks on the benchmark data sets and performance

TABLE 3 Nine-class beta-turn and nonturn class sizes in BT6376

Beta-turn types	Number of proteins	Number of turns	Number of nonturns	Turns/nonturns
Type I	6039	42 393	1 366 911	0.031
Type I′	2786	4353	747 596	0.005
Type II	4750	13 559	1 183 457	0.011
Type II′	1995	2643	545 300	0.004
Type IV	5950	38 201	1 360 907	0.028
Type VIa1	600	654	182 122	0.003
Type VIa2	177	188	56 761	0.003
Type VIb	914	1082	263 099	0.004
Type VIII	4257	10 111	1 114 707	0.009

comparison with existing methods are presented. To evaluate the performance of our tool DeepDIN, 5-fold cross-validation was used on all data sets.

3.1 | How input features affect DeepDIN performance

As we used four input features for our proposed DeepDIN architecture, it is important to quantitatively determine how much improvement the proposed DeepDIN can make by using single, some, or all of the features (see Table 4). We used type I beta-turns for experiments. This data set has 6039 proteins containing a total of 42 393 type I beta-turns. In each experiment, five rounds of 5-fold cross-validation were performed to take into consideration data variation and model variation. The running time for each experiment was around 15 to 17 hours.

Table 4 shows that the predicted shape string feature set, either used alone or used in combined with other features, can significantly improve the prediction performance. Notably, some combined features have better prediction results than those features used alone, which means these features are complementary. For instance, when the predicted shape strings were combined with the predicted secondary structures as the input, the prediction performance is much better than just using the predicted shape strings. Although both features are related to the protein secondary structures, they may capture different aspects of features for a protein. In Table 4, the row 12 (0.465 [±0.002]) and row 15 (0.469 [±0.002]) have only marginal difference in MCC values. This may be because the HHBlits profile information had been already utilized in the eight-state secondary structure prediction in MUFold-SS.

3.2 | Residue-level and turn-level two-class predictions on BT426

Table 5 shows experimental results of comparing DeepDIN with existing methods at residue level on benchmark BT426. Other than BetaTPred3,[7] all other tools only had residue-level predictions. At turn-level prediction, BetaTPred3[7] achieved MCC 0.43, while DeepDIN achieved 0.550 (±0.019). At both residue level and turn level, DeepDIN outperformed all existing methods. It is noted that in Ref. [52], the authors reported their turn-level MCC around 0.66 on benchmark BT426. However, they preprocessed the data set in a way that the negative (nonturn) samples were randomly selected when the ratio of positive to negative was 1:3,[52] which significantly favored MCC over the setting used by BetaTPred3 and DeepDIN, that is, using all the residues in the original BT426 data set with the ratio of positive to negative as high as 1:9. In addition, the study by Tang et al[52] did not have a downloadable software tool or web-server to compare with. Hence, we did not compare the performance of Tang et al[52] in this study.

TABLE 4 Effects of feature combinations on prediction performance in terms of MCC

Physicochemical	SS8	Shape string	HHBlits profile	MCC
x				0.186 (±0.003)
	x			0.321 (±0.004)
		x		0.372 (±0.004)
			x	0.326 (±0.002)
x	x			0.406 (±0.003)
x		x		0.400 (±0.002)
x			x	0.366 (±0.001)
	x		x	0.369 (±0.002)
	x	x		0.435 (±0.006)
		x	x	0.421 (±0.003)
	x	x	x	0.451 (±0.002)
x	x	x		0.465 (±0.002)
x		x	x	0.440 (±0.002)
x	x		x	0.416 (±0.002)
x	x	x	x	0.469 (±0.002)

Abbreviations: HHBlits, hidden Markov model (HMM)-HMM-based lightning; MCC, Matthew correlation coefficient.

TABLE 5 Residue-level prediction on BT426

Predictor	MCC
BTPred[19]	0.35
BetaTPred2[13,22]	0.43
Hu and Li[24]	0.47
NetTurnP[31]	0.50
BetaTPred3-Tweak[7]	0.50
BetaTPred3-7Fold[7]	0.51
BetaTPred3[7]	0.51
DeepDIN	0.647 (±0.016)

Abbreviations: DeepDIN, deep dense inception network; MCC, Matthew correlation coefficient.

3.3 | Turn-level nine-class prediction on BT6376

Table 6 shows the nine-class beta-turn prediction results on BT6376. The improvement of DeepDIN over BetaTPred3 ranges from 0.01 to 0.22 MCC. The average MCC results of DeepDIN for large class beta-

TABLE 6 Performance of nine-class beta-turn prediction evaluated at turn-level on BT6376

Beta-turn types	BetaTPred3 average MCC	DeepDIN average MCC
Type I	0.30	0.462 (±0.027)
Type I′	0.45	0.645 (±0.057)
Type II	0.35	0.569 (±0.050)
Type II′	0.33	0.564 (±0.061)
Type IV	0.20	0.292 (±0.016)
Type VIa1	0.33	0.395 (±0.023)
Type VIa2	0.25	0.262 (±0.121)
Type VIb	0.35	0.465 (±0.023)
Type VIII	0.17	0.234 (±0.015)

Abbreviations: DeepDIN, deep dense inception network; MCC, Matthew correlation coefficient.

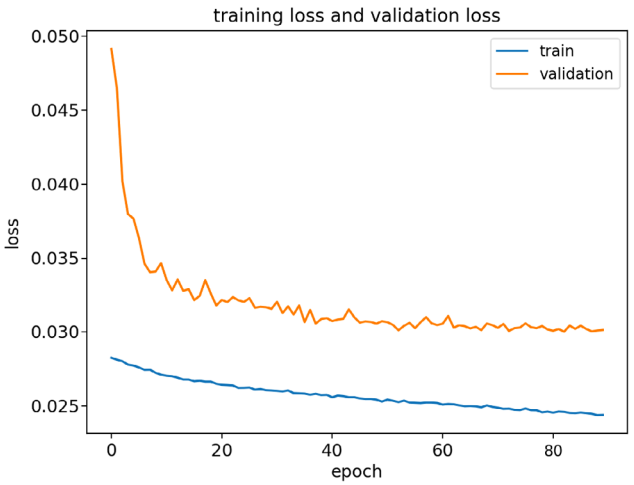

FIGURE 4 Training loss and validation loss curves of one of the models trained for type I beta-turn classification [Color figure can be viewed at wileyonlinelibrary.com]

turns such as I, I′, II, II′, IV, outperformed BetaTPred3 by at least 7%, which is a significant improvement. For some small classes such as VIa1 and Vib, the improvement is about 3%. DeepDIN performed comparably with BetaTPred3 on small class VIa2.

As the beta-turn data set is very imbalanced, during training, different class weights were calculated using the Scikit-learn toolbox[49] and then assigned to the loss function during the model training process. A 5-fold cross-validation evaluation was performed on each of the nine-class beta-turn classification tasks. The average experiment time ranged from 2 to 4 hours depending on the different amounts of data in each class. Figure 4 shows the training loss and validation loss curves for one of the models trained for type I beta-turn classification, as an example.

4 | CONCLUSION

There are many successful deep learning applications in protein bioinformatics,[53-55] including applications in modeling protein sequences to predict hierarchical labels[56] and deep learning with unbalanced data.[57] Here, a new deep neural network architecture DeepDIN was proposed for protein beta-turn prediction. Extensive experimental results show that DeepDIN obtained more accurate predictions than the best state-of-the-art methods and tools. Compared to previous machine-learning methods for protein beta-turn prediction, this work uses a more sophisticated, yet efficient, deep-learning architecture. There are several innovations in this work.

First, the proposed DeepDIN takes input of the entire protein sequence as the input feature, whereas all previous predictors relied on a fix-sized sliding window. DeepDIN is more effective and efficient in extracting long-range residue interactions, as it utilizes densely connected inception blocks to process local and nonlocal interactions of residues.

Second, we have implemented a tool called MUFold-BetaTurn, which utilizes the proposed DeepDIN architecture for beta-turn prediction. This tool is ready for the research community to use freely.

Third, this work quantitatively discovered how different input features affect the beta-turn prediction. For example, some features such as shape string and HHBlits profile can improve beta-turn classification effectively. In the future work, the small beta-turn classes still have room for further improvement. Also, those features can be useful for other turn prediction problems, such as gamma-turn prediction.[58]

Last but not least, we demonstrated a good example of the small imbalanced data set classification problem using balanced learning and transfer learning. The beta-turn data set is very imbalanced, where the ratio of positive samples over negative samples is about 1:9. Balanced learning and transfer learning were applied to overcome the problem. It is worth mentioning that transfer learning was originally applied to image recognition tasks, and here, we applied a similar method to training models for small beta-turn classification tasks. The pretrained network is effective in learning some more general beta-turn features; then the transfer learning technique can transfer the base network to some more specific models that can classify nine-class beta-turns. We have also demonstrated some techniques for tuning deep neural networks on small data classification problems, which may be useful in other areas of biological sequence analyses with imbalanced data sets, such as genomic analysis,[59] poly-signal identification,[60] post-translational modification prediction,[61] and so forth.

ACKNOWLEDGEMENTS

This work was partially supported by National Institutes of Health Grant R01-GM100701.

ORCID

Dong Xu https://orcid.org/0000-0002-4809-0514

REFERENCES

1. Dill KA, MacCallum JL. The protein-folding problem, 50 years on. *Science*. 2012;338(6110):1042-1046.

2. Zhou Y, Duan Y, Yang Y, Faraggi E, Lei H. Trends in template/fragment-free protein structure prediction. *Theor Chem Acc*. 2011;128(1):3-16.
3. Webb B, Sali A. Protein structure prediction. *Curr Protoc Bioinformatics*. 2014;1137:1-5.
4. Richardson JS. The anatomy and taxonomy of protein structure. *Advances in Protein Chemistry*. Vol 34. Cambridge, MA: Academic Press; 1981:167-339.
5. Milner-White EJ, Poet R. Loops, bulges, turns and hairpins in proteins. *Trends Biochem Sci*. 1987;12:189-192.
6. Rose GD, Glerasch LM, Smith JA. Turns in peptides and proteins. *Advances in Protein Chemistry*. Vol 37. Cambridge, MA: Academic Press; 1985:1-109.
7. Singh H, Singh S, Raghava GP. In silico platform for predicting and initiating β-turns in a protein at desired locations. *Proteins*. 2015;83(5):910-921.
8. Lewis PN, Momany FA, Scheraga HA. Chain reversals in proteins. *Biochim Biophys Acta*. 1973;303(2):211-229.
9. Chou KC. Prediction of tight turns and their types in proteins. *Anal Biochem*. 2000;286(1):1-6.
10. Hutchinson EG, Thornton JM. A revised set of potentials for β-turn formation in proteins. *Protein Sci*. 1994;3(12):2207-2216.
11. Li SZ, Lee JH, Lee W, Yoon CJ, Baik JH, Lim SK. Type I β-turn conformation is important for biological activity of the melanocyte-stimulating hormone analogues. *Eur J Biochem*. 1999;265(1):430-440.
12. Ramírez-Alvarado M, Blanco FJ, Niemann H, Serrano L. Role of β-turn residues in β-hairpin formation and stability in designed peptides1. *J Mol Biol*. 1997;273(4):898-912.
13. Kaur H, Raghava GP. BetaTPred: prediction of β-turns in a protein using statistical algorithms. *Bioinformatics*. 2002;18(3):498-499.
14. Chou PY, Fasman GD. Conformational parameters for amino acids in helical, β-sheet, and random coil regions calculated from proteins. *Biochemistry*. 1974;13(2):211-222.
15. Chou PY, Fasman GD. Prediction of beta-turns. *Biophys J*. 1979;26(3):367-383.
16. Zhang CT, Chou KC. Prediction of β-turns in proteins by 1-4 and 2-3 correlation model. *Biopolymers Original Res Biomolecules*. 1997;41(6):673-702.
17. Chou KC. Prediction of beta-turns in proteins. *J Pept Res*. 1997;49(2):120-144.
18. McGregor MJ, Flores TP, Sternberg MJ. Prediction of β-turns in proteins using neural networks. *Protein Eng Design Selection*. 1989;2(7):521-526.
19. Shepherd AJ, Gorse D, Thornton JM. Prediction of the location and type of β-turns in proteins using neural networks. *Protein Sci*. 1999;8(5):1045-1055.
20. Kim S. Protein β-turn prediction using nearest-neighbor method. *Bioinformatics*. 2004;20(1):40-44.
21. Fuchs PF, Alix AJ. High accuracy prediction of β-turns and their types using propensities and multiple alignments. *Proteins*. 2005;59(4):828-839.
22. Kaur H, Raghava GP. A neural network method for prediction of β-turn types in proteins using evolutionary information. *Bioinformatics*. 2004;20(16):2751-2758.
23. Kirschner A, Frishman D. Prediction of β-turns and β-turn types by a novel bidirectional Elman-type recurrent neural network with multiple output layers (MOLEBRNN). *Gene*. 2008;422(1):22-29.
24. Hu X, Li Q. Using support vector machine to predict β-and γ-turns in proteins. *J Comput Chem*. 2008;29(12):1867-1875.
25. Zheng C, Kurgan L. Prediction of beta-turns at over 80% accuracy based on an ensemble of predicted secondary structures and multiple alignments. *BMC Bioinformatics*. 2008;9(1):430.
26. Jones DT. Protein secondary structure prediction based on position-specific scoring matrices1. *J Mol Biol*. 1999;292(2):195-202.
27. Cole C, Barber JD, Barton GJ. The Jpred 3 secondary structure prediction server. *Nucleic Acids Res*. 2008;36(suppl 2):W197-W201.
28. Montgomerie S, Sundararaj S, Gallin WJ, Wishart DS. Improving the accuracy of protein secondary structure prediction using structural alignment. *BMC Bioinformatics*. 2006;7(1):301.
29. Montgomerie S, Cruz JA, Shrivastava S, Arndt D, Berjanskii M, Wishart DS. PROTEUS2: a web server for comprehensive protein structure prediction and structure-based annotation. *Nucleic Acids Res*. 2008;36(suppl 2):W202-W209.
30. Kountouris P, Hirst JD. Predicting β-turns and their types using predicted backbone dihedral angles and secondary structures. *BMC Bioinformatics*. 2010;11(1):407.
31. Petersen B, Lundegaard C, Petersen TN. NetTurnP–neural network prediction of beta-turns by use of evolutionary information and predicted protein sequence features. *PLoS One*. 2010;5(11):e15079.
32. LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*. 2015;521(7553):436-444.
33. Fang C, Shang Y, Xu D. MUFOLD-SS: new deep inception-inside-inception networks for protein secondary structure prediction. *Proteins Struct Funct Bioinformat*. 2018;86(5):592-598.
34. Fang C, Shang Y, Xu D. A new deep neighbor residual network for protein secondary structure prediction. *IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, Boston, MA; 2017:66-71.
35. Fang C, Shang Y, Xu D. Prediction of protein backbone torsion angles using deep residual inception neural networks. *IEEE/ACM Trans Comput Biol Bioinform*. 2018; 16:1020-1028.
36. Szegedy C, Ioffe S, Vanhoucke V, Alemi AA. Inception-v4, inception-resnet and the impact of residual connections on learning. *AAAI*. 2017;4:12.
37. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. *Proc IEEE Conf Comput Vision Pattern Recog*. 2016;770-778.
38. Remmert M, Biegert A, Hauser A, Söding J. HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nat Methods*. 2012;9(2):173-175.
39. Zhou T, Shu N, Hovmöller S. A novel method for accurate one-dimensional protein structure prediction based on fragment matching. *Bioinformatics*. 2009;26(4):470-477.
40. Heffernan R, Yang Y, Paliwal K, Zhou Y. Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility. *Bioinformatics*. 2017;33(18):2842-2849.
41. Ison RE, Hovmoller S, Kretsinger RH. Proteins and their shape strings. *IEEE Eng Med Biol Mag*. 2005;24(3):41-49.
42. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ. Densely connected convolutional networks. *CVPR*. 2017;1(2):3.
43. Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift. *arXiv preprint*. arXiv:1502.03167. 2015.
44. Radford A, Metz L, Chintala S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint*. arXiv:1511.06434. 2015.
45. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res*. 2014;15(1):1929-1958.
46. Kingma DP, Ba J. Adam: a method for stochastic optimization. *arXiv preprint*. arXiv:1412.6980. 2014.
47. Han H, Wang WY, Mao BH. Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. *International Conference on Intelligent Computing*. Berlin, Heidelberg: Springer; 2005:878-887.
48. Tomek I. Two modifications of CNN. *IEEE Trans Syst Man Cybernet*. 1976;6:769-772.
49. Pedregosa F, Varoquaux G, Gramfort A, et al. Machine learning in python. *J Mach Learn Res*. 2011;12(Oct):2825-2830.
50. Raina R, Battle A, Lee H, Packer B, Ng AY. Self-taught learning: transfer learning from unlabeled data. *Proceedings of the 24th*

International Conference on Machine Learning. New York, NY: ACM; 2007:759-766.
51. Guruprasad K, Rajkumar S. Beta-and gamma-turns in proteins revisited: a new set of amino acid turn-type dependent positional preferences and potentials. *J Biosci*. 2000 Jun;25(2):143-156.
52. Tang Z, Li T, Liu R, et al. Improving the performance of β-turn prediction using predicted shape strings and a two-layer support vector machine model. *BMC Bioinformatics*. 2011 Dec;12(1):283.
53. Min S, Lee B, Yoon S. Deep learning in bioinformatics. *Brief Bioinform*. 2017;18(5):851-869.
54. Angermueller C, Pärnamaa T, Parts L, Stegle O. Deep learning for computational biology. *Mol Syst Biol*. 2016;12(7):878.
55. Li Y, Huang C, Ding L, Li Z, Pan Y, Gao X. Deep learning in bioinformatics: introduction, application, and perspective in big data era. *arXiv preprint. arXiv*:1903.00342. 2019.
56. Li Y, Wang S, Umarov R, et al. DEEPre: sequence-based enzyme EC number prediction by deep learning. *Bioinformatics*. 2017;34(5):760-769.
57. Umarov R, Kuwahara H, Li Y, Gao X, Solovyev V. Promoter analysis and prediction in the human genome using sequence-based deep learning models. *Bioinformatics*. 2019;1:8.
58. Pham TH, Satou K, Ho TB. Support vector machines for prediction and analysis of beta and gamma-turns in proteins. *J Bioinform Comput Biol*. 2005 Apr;3(02):343-358.
59. Zou J, Huss M, Abid A, Mohammadi P, Torkamani A, Telenti A. A primer on deep learning in genomics. *Nat Genet*. 2018;51(1):12-18.
60. Xia Z, Li Y, Zhang B, et al. DeeReCT-PolyA: a robust and generic deep learning method for PAS identification. *Bioinformatics*. In press.
61. Chen Z, Liu X, Li F, et al. Large-scale comparative assessment of computational predictors for lysine post-translational modification sites. *Brief Bioinform*. In press.

How to cite this article: Fang C, Shang Y, Xu D. A deep dense inception network for protein beta-turn prediction. *Proteins*. 2020;88:143–151. https://doi.org/10.1002/prot.25780