Combining Deep Neural Networks for Protein Secondary Structure Prediction

SHUSEN ZHOU, HAILIN ZOU, CHANJUAN LIU, MUJUN ZANG, AND TONG LIU

School of Information and Electrical Engineering, Ludong University, Yantai 264025, China

Corresponding author: Shusen Zhou (zhoushusen@ldu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61300155.

ABSTRACT By combining convolutional neural networks (CNN) and long short term memory networks (LSTM) into the learning structure, this paper presents a supervised learning method called combining deep neural networks (CDNN) for protein secondary structure prediction. First, we use multiple convolutional neural networks with different number of layers and different size of filters to extract the protein secondary structure features. Second, we use bidirectional LSTM to extract features continually based on the raw features and the features which extracted by CNNs. Third, a fully connected dense layer is used to map the extracted features by LSTM to the different protein secondary structure classes. CDNN architecture is trained by RMSProp optimizer based on the cross entropy error between protein secondary structure labels and dense layer’s outputs. CDNN not only inherits the abstraction ability of CNN and sequence data process ability of LSTM, but also demonstrates the attractive classification ability for handling protein secondary structure data. The empirical validation on two protein secondary structure prediction datasets demonstrates the effectiveness of CDNN method.

INDEX TERMS Deep learning, supervised learning, prediction, protein secondary structure.

I. INTRODUCTION

A protein’s structure determines its function [1]. Understanding complex dependency between protein structure and sequence is one of the greatest challenges in computational biology [2]. One subproblem of protein structure prediction is the prediction of protein secondary structure [3], which has been extensively studied with machine learning approaches.

Recently, the use of deep neural networks proved to be effective and significantly improved previous accuracy on the secondary structure prediction problem [2]–[14]. Application of deep learning in bioinformatics to gain insight from data has been emphasized in both academia and industry [15].

Convolutional neural networks (CNN) are designed to process data that come in the form of multiple arrays, which have brought about breakthroughs in processing image, video, speech and audio. Recurrent neural networks are a generalization of feed forward neural networks that naturally handle sequential data [5], which process an input sequence one elements at a time, maintaining in their hidden units a ‘state vector’ that implicitly contains information about the history of all the past elements of the sequence, and have shone light on sequential data. RNNs with long short term memory cells (LSTM) augment the network with an explicit memory, have subsequently proved to be more effective than conventional RNNs [16]. In this paper, a novel supervised learning method called combining deep neural networks (CDNN) is proposed for protein secondary structure prediction, which is based on the CNN and LSTM. Some favorable properties of CDNN are shown below:

1) CDNN gets the capability of extracting protein secondary structure features based on combined CNNs. It utilizes multiple CNNs with different number of layers and different size of filters. CNNs with different number of layers can extract the features with different levels. CNNs with different size of filters can extract the features with different number of neighbouring amino acids. The combined features extracted by these CNNs preserve the information well with different levels and different number of neighbouring amino acids.

2) CDNN gets the capability of extracting protein secondary structure features based on bidirectional LSTM. We use a bidirectional recurrent neural networks with LSTM cells to extract the amino acid sequential information effectively based on the raw features and the features which extracted by CNNs.

3) CDNN uses a fully connected layer to map the extracted features by LSTM to the different protein
TABLE 1. The 8-class to 3-class mappings of protein secondary structure classes.

| 8-class | 3-class | Name     |
|---------|---------|----------|
| H       | H       | α-helix  |
| E       | E       | β-strand |
| L       | C       | loop or irregular |
| T       | C       | β-turn   |
| S       | C       | bend     |
| G       | H       | 3_10-helix |
| B       | E       | β-bridge |
| I       | C       | π-helix  |

secondary structure classes. CDNN shows impressive prediction results on protein secondary structure dataset with RMSProp optimizer based on the cross entropy error between protein secondary structure labels and LSTM’s outputs.

II. RELATED WORK

Protein secondary structure prediction began in 1951, knowing secondary structures provide an approximate idea about overall structural categories [3]. By convention, three regular secondary structure classes are used for protein secondary structure prediction. Kabsch and Sander [17] developed a DSSP algorithm to classify secondary structure into 8 fine-grained classes [6], the 8-class DSSP output is typically mapped to 3-class. The 8-class to 3-class mappings of protein secondary structure classes are shown in Table 1. Most secondary structure prediction studies have been focusing on coarse-grained 3-class secondary structure prediction. In this paper, we focus on fine-grained 8-class secondary structure prediction, because it reveals more structural details and more challenging [2].

Many machine learning methods have been proposed for protein secondary structure prediction [3]. Rost and Sander used a neural network algorithm to increase in both the accuracy and quality of secondary structure predictions [18]. Jones used a two-stage neural network to predict protein secondary structure based on the position specific scoring matrices [19]. Hua and Sun introduced a new method of protein secondary structure prediction which is based on the theory of support vector machines [20]. Ward et al. proposed a support vector machines based method to predict protein secondary structure [21]. Aydin et al. further refined and extended the hidden semi-Markov model for protein secondary structure prediction [22]. Dor and Zhou established and optimized an integrated neural networks for predicting structural properties of proteins [23]. Yao et al. reported a new method of probabilistic nature for protein secondary structure prediction, based on dynamic Bayesian networks [24]. Wang et al. presented a new probabilistic method for 8-class secondary structure prediction using conditional neural fields [25]. Dongardive and Abraham proposed an optimized parameter set for protein secondary structure prediction using three-layer feed forward back propagation neural network [26]. Xie et al. proposed a new method based on an improved fuzzy support vector for the prediction of the secondary structure of proteins [27]. Torrisi et al. presented Porter 5, one of the best performing ab initio secondary structure predictor, significantly outperforming all the most recent ab initio predictors [28]. Buchan and Jones presented a web server offering a range of predictive methods to the bioscience community for 20 years [29].

Recently, several works have been proposed for protein secondary structure prediction with deep learning, and have got competitive performance. Zhou and Troyanskaya presented a supervised generative stochastic network based method to predict protein secondary structure with deep hierarchical representations [2]. Spencer et al. developed an secondary structure predictor that makes use of the position-specific scoring matrix and deep learning network architectures [4]. Sonderby and Winther used a bidirectional recurrent neural network with long short term memory cells for prediction of protein secondary structure [5]. Wang et al. presented an integration of conditional neural fields and shallow neural networks for protein secondary structure prediction [6]. Li et al. used an ensemble of ten independently trained models, each comprised of a multi-scale convolutional layer followed by three stacked bidirectional recurrent layers, to predict protein secondary structure from integrated local and global contextual features [7]. Heffernan et al. employed long short-term memory bidirectional recurrent neural networks to the prediction of protein structural properties [8]. Wang et al. proposed a deep recurrent encoder-decoder networks to solve secondary structure prediction problem [9]. Busia et al. created a novel chained convolutional architecture with next-step conditioning for improving performance on protein sequence prediction problems. They modeled the dependencies between secondary structure labels by conditioning the current prediction on the previous structure labels in addition to the current input [10]. Fang et al. used a new deep neural network architecture, named the deep inception-inside-inception network, for protein secondary structure prediction [11]. Heffernan et al. showed a single-sequence-based prediction method employing a long short-term bidirectional recurrent neural networks for protein structure prediction [12]. Hanson et al. leveraged an ensemble of LSTM-BRNN and ResNet models, together with predicted residue-residue contact maps, to continue the push towards the attainable limit of prediction for 3-class and 8-class secondary structure [13]. Klausen et al. used an architecture composed of convolutional and long short-term memory neural networks trained on solved protein structures, to predict the most important local structural features [14].

III. COMBINING DEEP NEURAL NETWORKS

Protein secondary structure is predicted based on amino acid sequence. While CNN can extract the features of data layer by layer effectively, it cannot be expected to accurately extract amino acid sequence data information. On the other hand, LSTM can handle sequential data effectively. Motivated by this observation, we combine CNN and LSTM in one deep
architecture, and propose combining deep neural networks (CDNN) method to address the protein secondary structure prediction problem with supervised learning:

1) We propose a novel CDNN architecture which integrates the abstraction ability of CNN and LSTM in one deep architecture.

2) We firstly combine multiple CNNs with different number of layers and different size of filters, to extract the different levels and different adjacent domains information of amino acid sequence.

3) We then use a LSTM layer to process the output of CNNs and raw features of amino acid sequence continually. These combined features contain more information, which can be used to train bigger network effectively.

4) At last, we use a dense layer to process the output of LSTM continually. Then the extracted features are mapped to the label layer.

A. ARCHITECTURE OF COMBINING DEEP NEURAL NETWORKS

The protein secondary structure prediction dataset composed of many amino acid sequences labeled with secondary structure. Sequences and structures were downloaded from PDB and annotated with the DSSP program [5], we use the original 8-class DSSP output. Each amino acid is represented as a vector of features \( \mathbf{x} \). Then the protein dataset is represented as a matrix:

\[
X = \begin{bmatrix} x^1, x^2, \ldots, x^L \end{bmatrix}
\]

where \( x = [x_1, x_2, \ldots, x_{N_0}]' \)

\( L \) is the number of amino acids, \( N_0 \) is the number of features in the amino acid.

Let \( Y \) be a set of protein secondary structure labels corresponding to \( L \) labeled training amino acids and is denoted as:

\[
Y = \begin{bmatrix} y^1, y^2, \ldots, y^L \end{bmatrix}
\]

where \( y = [y_1, y_2, \ldots, y_C]' \)

\( y_j = \begin{cases} 
1 & \text{if } x \in j^{th} \text{ class} \\
0 & \text{if } x \notin j^{th} \text{ class} 
\end{cases} \)

\( C \) is the number of protein secondary structure classes.

The deep architecture is designed to seek the mapping function \( X \rightarrow Y \) based on the \( L \) labeled amino acids, then we can determine \( y \) for a new inputting amino acid \( x \). We design a supervised learning algorithm CDNN using deep learning techniques to solve this problem. The deep architecture of CDNN is shown in Fig. 1, a fully combining deep architecture with input layer, CNN layer, LSTM layer, dense layer, and one label layer. The input layer has \( N_0 \) units, equal to the number of features in sample amino acid \( x \). One protein with \( L \) amino acids will be input to the deep architecture at a time to extract the sequence information. The label layer has \( C \) different units, equal to number of protein secondary structure classes of label vector \( y \). There are multiple CNNs with different number of layers and different size of filters in CNN layer. The output of CNN layer and input layer are combined and input to the LSTM layer, which can provide more information with different levels and different adjacent domains information in amino acid sequence. Bidirectional LSTM is used in LSTM layer, which can be used to extract the sequence information effectively by combining a LSTM that moves forward through time beginning from the start of the sequence with another LSTM that moves backward through time beginning from the end of the sequence. Then the forward and backward outputs are input to the dense layer, and the outputs of dense layer are mapped to the label layer. The seeking of mapping function \( X \rightarrow Y \), is transformed to finding the parameter space \( W \) for CDNN. We will detail introduce the CDNN method in the following sections.

B. CONVOLUTIONAL NEURAL NETWORKS LAYER

In CNN layer, we use 3 different level (1 layer, 2 layers and 3 layers) CNNs to extract amino acid features, then combined with raw features, output to LSTM layer.
The architecture of 1 layer CNNs can be seen in Fig. 2, the number of nodes in convolution kernel is 3, 5, 7, 9 and 11 in these 5 parts, which can extract the amino acid features effectively with different size of windows.

The function of these CNNs is feature extraction, each part including \( N_1 \) convolution kernels. Each convolution kernel including \( f \) weight coefficients and 1 bias, similarly with neurons of a feedforward neural network. For each input \( x \) and its neighboring amino acids, convolution kernels take the sum of matrix element multiplication in turn. After convolution, weighted sum the obtained features and add the bias:

\[
h^f_{il}(X) = \sum_{k=1}^{N_0} \sum_{j=1}^{f} \left[ h^0_{(i+j)k}(X) w^f_{kj} \right] + b^f_l\\
\]

where \( h^0(X) \) and \( h^1(X) \) are the input and output of convolution, \( h^0_{(i+j)k}(X) \) equal to the \( k^{th} \) feature of \( (i+j)^{th} \) amino acid in input \( X \). h^f_{il}(X) indicate the \( l^{th} \) feature of \( i^{th} \) amino acid in output \( h^1(X) \), the used convolution kernels including \( f \) weight coefficients. \( w^f_{kj} \) and \( b^f_l \) are the model parameters: \( w^f_{kj} \) is convolution kernel between \( k^{th} \) feature of input \( h^0(X) \) and \( l^{th} \) feature of output \( h^1(X) \), \( b^f_l \) is the \( l^{th} \) bias of \( 1^{th} \) CNN layer. \( N_0 \) is the number of units in \( h^0(X) \). \( N_1 \) is the number of units in \( h^1(X) \).

Then the output \( h^1(X) \) is activated by ReLU function:

\[
a^f(X) = \text{ReLU} \left( h^f(X) \right) \tag{7}\\
\]

At last, we use batch-normalized function to avoid model overfitting:

\[
da^f(X) = \text{BN} \left( a^f(X) \right) \tag{8}\\
\]

We set the value of \( f \) equal to 3, 5, 7, 9, 11 and get 5 outputs \( a^{1.3}(X), a^{1.5}(X), a^{1.7}(X), a^{1.9}(X), a^{1.11}(X) \).

The 2 layers CNNs use one more convolution, activation and batch-normalization operations than 1 layer CNNs, the number of nodes in convolution kernel is 3, 5, 7, 9 and 11 in these 5 parts too. For each input \( x \) and its neighboring amino acids, two times of convolution, activation and batch-normalization were carried out successively. After \( a^f(X) \) is calculated based on Eq. 8, the second layer calculation is below:

\[
h^2_{il}(X) = \sum_{k=1}^{N_1} \sum_{j=1}^{f} \left[ d^f_{(i+j)k}(X) w^2_{kj} \right] + b^2_l\\
\]

where \( d^f(X) \) and \( h^2(X) \) are the input and output of convolution. \( h^2_{il}(X) \) indicate the \( l^{th} \) feature of \( i^{th} \) amino acid in output \( h^2(X) \), the used convolution kernels including \( f \) weight coefficients. \( w^2_{kj} \) is convolution kernel between \( k^{th} \) feature of input \( d^f(X) \) and \( l^{th} \) feature of output \( h^2(X) \), \( b^2_l \) is the \( l^{th} \) bias of \( 2^{th} \) CNN layer. \( N_2 \) is the number of units in \( h^2(X) \).

\[
a^2(X) = \text{ReLU} \left( h^2(X) \right) \tag{9}\\
da^2(X) = \text{BN} \left( a^2(X) \right) \tag{10}\\
\]

We set the value of \( f \) equal to 3, 5, 7, 9, 11 and get 5 outputs \( a^{2.3}(X), a^{2.5}(X), a^{2.7}(X), a^{2.9}(X), a^{2.11}(X) \).

The 3 layers CNNs use one more convolution, activation and batch-normalization operations than 2 layers CNNs, the number of nodes in convolution kernel is 3, 5, 7, 9 and 11 in these 5 parts too. For each input \( x \) and its neighboring amino acids, three times of convolution, activation and batch-normalization were carried out successively. After \( a^2(X) \) is calculated based on Eq. 11, the third layer calculation
is below:

\[
    h_{il}^3 (X) = \sum_{k=1}^{N_3} \sum_{j=1}^{f} \left[ a_{(i+j)k}^2 (X) w_{kj}^3 \right] + b_i^3 \\
    l \in \{0, 1, \ldots, N_3\} 
\]

where \(a^2 (X)\) and \(h^3 (X)\) are the input and output of convolution. \(h_{il}^3 (X)\) indicate the \(i^{th}\) feature of \(i^{th}\) amino acid in output \(h^3 (X)\), the used convolution kernels including \(w\) weight coefficients. \(w_{kj}^3\) is convolution kernel between \(k^{th}\) feature of input \(d^2 (X)\) and \(l^{th}\) feature of output \(h^3 (X)\). \(b_i^3\) is the \(l^{th}\) bias of \(3^{rd}\) CNN layer. \(N_3\) is the number of units in \(h^3 (X)\).

\[
    a^3 (X) = ReLU \left( h^3 (X) \right) 
\]

\[
    d^3 (X) = BN \left( a^3 (X) \right) 
\]

We set the value of \(f\) equal to 3, 5, 7, 9, 11 and get 5 outputs \(d^{3,3} (X), d^{3,5} (X), d^{3,7} (X), d^{3,9} (X), d^{3,11} (X)\).

After the feature extraction of 1 layer, 2 layers and 3 layers CNNs, we combine the features of these CNNs outputs and raw input \(X\), which can be indicated as:

\[
    e = X, d^{1,3}, d^{1,5}, d^{1,7}, d^{1,9}, d^{1,11}, d^{2,3}, d^{2,5}, d^{2,7}, d^{2,9}, d^{2,11}, d^{3,3}, d^{3,5}, d^{3,7}, d^{3,9}, d^{3,11} 
\]

C. LONG SHORT TERM MEMORY LAYER

In LSTM layer, we use bidirectional LSTM to extract amino acid features. Bidirectional LSTM is combined by a LSTM that moves forward through time beginning from the start of the sequence with another LSTM that moves backward through time beginning from the end of the sequence, then the forward and backward outputs are combined and output to the dense layer.

\[
    g_t (X) = \left[ \tilde{e}_t (X) \oplus e_t (X) \right] 
\]

where \(\tilde{e}_t (X) = LSTM (e_t (X), \quad e_{t-1} (X))\) denote the feature representation on \(t^{th}\) position extracted by LSTM based on first \(t-1\) amino acid features. \(e_t (X) = LSTM (e_t (X), \quad e_{t+1} (X))\) denote the feature representation on \(t^{th}\) position extracted by LSTM based on last \(L-t\) amino acid features. The number of units in hidden layer of LSTM is denoted as \(N_d\).

D. DENSE LAYER

At last, we map the features extracted by Eq. 16 to the dense layer of the deep architecture, which has \(C\) nodes, equal to the number of amino acid classes.

\[
    o_t (X) = \sum_{s=1}^{N} w_{st} g_s (X) + b_t 
\]

where \(w_{st}\) is the weight between unit \(s\) in the layer \(g (X)\) and unit \(t\) in the layer \(o (X)\). \(b_t\) is the \(t^{th}\) bias of layer \(o (X)\). \(g_s (X)\) is the output of Eq. 16, corresponding to \(s^{th}\) feature of a amino acid. \(N\) is the number of features in \(g (X)\).

Then we can train the deep architecture, use \(L\) labeled amino acid to adjust the parameter space \(W\) for better classification ability. This task is formulated as an optimization problem:

\[
    \arg \min_W f (o (X), Y) 
\]

where

\[
    f (o (X), Y) = \sum_{i=1}^{L} \sum_{j=1}^{C} T \left( o (x_i^j) \right) y_i^j 
\]

\(T\) denote the cross entropy loss function. We use RMSProp optimizer to train the whole deep architecture based on loss function \(T\), and use L2 regularization method to prevent overfitting.

E. COMBINING DEEP NEURAL NETWORKS ALGORITHM

In CDNN algorithm, we use the amino acid data \(X\), and its corresponding labels \(Y\) at the same time to train the deep architecture, the parameters are randomly initialized with normal distribution. All the proteins in the dataset are used to train the CDNN with supervised learning. The number of units in every hidden layer of CNN \(N_1, N_2, N_3\), and number of units in hidden layer of LSTM \(N_d\) are set manually based on the dimension of the input data and the size of dataset.

Then we can determine the label of new amino acids based on the mapping results of CDNN deep architecture:

\[
    \hat{j} = \arg \max_j o(x) 
\]

IV. EXPERIMENTS

We report the experimental setup and conduct several experiments to compare CDNN with some other related methods firstly. Then we compare our method CDNN with other state-of-the-art protein secondary structure prediction methods. At last, we demonstrate the performance of our proposed method with different number of units in every hidden layer of CNNs, and the performance of our proposed method with different number of units in hidden layer of LSTM.

A. EXPERIMENTAL SETUP

We evaluate the performance of CDNN method using two protein secondary structure prediction datasets. The first dataset is CullPDB (generated on 2013.09.08) [30], which is commonly used for evaluating structure prediction algorithm. The second dataset is CB513 [31], which is used for testing. These two datasets are preprocessed by Zhou and Troyanskaya [2]. CullPDB has 6128 non-homologous sequences, which is further filtered such that no sequences has more than 50% identity with the CB513 dataset, then we have 5278 proteins for training and 256 proteins for testing. CB513 dataset contains 514 proteins for testing. CDNN is trained by 5278 proteins in CullPDB dataset, and test by 256 proteins in CullPDB dataset and CB513 dataset separately.
Similar with Zhou and Troyanskaya [2], we focus on 8-class secondary structure prediction as this is a more challenging problem than 3-class prediction and reveals more structural information. Each amino acid is encoded as a 42 dimensional vector, 21 dimensions (20 proteinogenic amino acids and 1 dimension reserved for non-standard/unknown residues) for orthogonal encoding and 21 dimensions for sequence profiles [5]. Evolutionary information as position-specific scoring matrix (PSSM) scores are used as sequence profiles, which are generated by running PSI-BLAST against UniRef90 (release date: 2014-05-14) database with inclusion threshold 0.001 and 3 iterations, then transformed by the sigmoid function to 0-1 range [2].

We compare the classification performance of CDNN with eight representative protein secondary structure prediction methods, i.e., deep supervised and convolutional generative stochastic networks (ICML2014) [2], long short term memory networks (LSTM large) [5], conditional neural fields (RaptorX-SS8) [25], deep convolutional neural fields (DeepCNF-SS) [6], secondary structure recurrent encoder-decoder networks (SSREDNs) [9], cascaded convolutional and recurrent neural networks (DCRNN) [7], deep convolutional neural networks (DCNN) [10], and protein secondary structure prediction using deep inception-inside-inception networks (MUFold-SS) [11].

B. PERFORMANCE OF COMBINING DEEP NEURAL NETWORKS

The architecture of CDNN model is mainly determined by the following 3 factors: (i) the number of units in every hidden layer of CNNs $N_1, N_2, N_3$; (ii) the number of units in hidden layer of LSTM $N_4$; (iii) the number of weight coefficients in convolution kernels of CNNs $f$, which control the number of processed neighboring amino acids. We fix $f$ equal to 3, 5, 7, 9, 11 respectively, because the average length of an alpha helix is around 11 residues. Moreover, the more adjoin the amino acids, the more information there is. So the nearest domains information of amino acid sequence is extracted repeatedly with these different number of weight coefficients in convolution kernels of CNNs. In this experiment, $N_1 = N_2 = N_3 = 100$, $N_4 = 1000$. There are 42 features for each amino acid, so the combined $e(X)$ has 1542 $(42 \times 100^3 + 3)$ features. The output of LSTM $g(X)$ has 2000 features. Although the raw features are rarely, through the combination of multiple CNNs, we get many features with different levels and different adjacent domains information in amino acid sequence. So we can use more hidden units in LSTM to extract the feature continually and effectively. CDNN is implemented based on TensorFlow using the Python language. RMSProp Optimizer with learning rate 0.001 is used for training model parameters. The CDNN network is trained for 50 epochs.

The 8-class prediction accuracies for two datasets and nine methods with supervised learning are shown in Table 2. Zhou and Troyanskaya [2] reported the results of ICML2014. Sonderey and Winther [5] reported the results of LSTM large method. Wang et al. [25] reported the results of RaptorX-SS8 method. Wang et al. [6] reported the results of DeepCNF-SS method. Wang et al. [9] reported the results of SSREDNs method. Li and Yu [7] reported the results of DCRNN method. Busia and Jaitly [10] reported the results of DCNN method. Fang et al. [11] reported the results of MUFold-SS method. The results of LSTM large, DCRNN, DCNN and MUFold-SS methods on CullPDB dataset are empty, because these corresponding papers did not report them. CDNN is the proposed method.

The results in Table 2 indicate that the performance of CDNN is competitive with other methods. CDNN gets the best result on CullPDB dataset, and is just slightly worse than DCRNN, DCNN and MUFold-SS methods. CDNN method is the combination of CNN and LSTM, all the experiment results on two dataset for CDNN method are better than LSTM large method, which prove the effectiveness of our proposed combination method. This could be contributed by: First, CDNN uses a new deep architecture, which combines the feature extraction ability of CNN and LSTM, and train the architecture with the cross entropy loss function to maximize the separability among different classes; Second, CDNN uses multiple CNNs to extract the features effectively with different levels and different adjacent domains information in amino acid sequence, which can train large LSTM networks and improve the classification performance of deep architecture; Third, CDNN uses CNNs to extract the features firstly, and uses LSTM to extract the features and sequence information continuously. The close cooperation of CNN and LSTM improves the classification ability of CDNN.

The 8-class prediction performance for individual secondary structure classes of CB513 are shown in Table 3. We achieved high accuracy for three major classes H, E and L. The samples of these three classes contain 73.25 percent of

### Table 2. The 8-class prediction accuracy for two datasets and nine methods.

| Methods        | CullPDB | CB513 |
|----------------|---------|-------|
| ICML2014       | 72.1    | 66.4  |
| LSTM large     | -       | 67.4  |
| RaptorX-SS8    | 69.7    | 64.9  |
| DeepCNF-SS     | 75.2    | 68.3  |
| SSREDNs        | 73.1    | 68.2  |
| DCRNN          | -       | 69.7  |
| DCNN           | -       | 70.0  |
| MUFold-SS      | 76.1    | 69.5  |

### Table 3. The 8-class prediction performance for individual secondary structure classes of CB513.

| 8-class | Accuracy | Frequency |
|---------|----------|-----------|
| H       | 93.3     | 0.30858   |
| E       | 80.5     | 0.21254   |
| L       | 66.2     | 0.21141   |
| T       | 46.8     | 0.11813   |
| S       | 29.3     | 0.09811   |
| G       | 30.5     | 0.05695   |
| B       | 3.3      | 0.01393   |
| I       | 0.0      | 0.00035   |
TABLE 4. The 3-class prediction accuracy for two datasets and five methods.

| Methods     | CallPDB | CB513 |
|-------------|---------|-------|
| RaptorX-SS8 | 81.2    | 78.3  |
| DeepCNF-SS  | **85.4** | 82.3  |
| SSREDNs     | 84.2    | **82.9** |
| MUFold-SS   | -       | **82.9** |
| CDNN        | 84.8    | 82.1  |

all samples. Prediction for less frequent classes S, G and B are difficult, because of limited number of training samples in these classes. Class I is extremely rare, we just encounter 0.035 percent in the test set, so it is hard to train the classifier on its prediction. Through the frequency of these eight classes in Table 3, we can find out the significantly unbalanced label problem for secondary structure classification. The prediction performance of classes with more samples are better than those with less samples. We did not consider this unbalanced label problem in CDNN method. In future, we will make more effort to better identify these less frequent classes.

Although we focus on 8-class secondary structure prediction, we also report the performance of CDNN against other state-of-the-art methods for 3-class secondary structure prediction. ICML2014, LSTM large, DCRNN and DCNN methods did not report their performance for 3-class secondary structure prediction, so we just compare with RaptorX-SS8, DeepCNF-SS, SSREDNs, and MUFold-SS methods. The 3-class prediction accuracy for two datasets and five methods with supervised learning are shown in Table 4. The results of RaptorX-SS8, DeepCNF-SS, SSREDNs, and MUFold-SS methods are reported by these corresponding papers [6], [9], [11], [25]. With the same network architecture and parameter settings, we report the results of CDNN. The results in Table 4 indicate that the performance of CDNN is competitive with other methods for 3-class secondary structure prediction too. The performance of DeepCNF-SS, SSREDNs, and MUFold-SS is just slightly better than our proposed CDNN for 3-class secondary structure prediction.

C. EXPERIMENTS WITH DIFFERENT NUMBER OF UNITS IN CNNs

In CDNN architecture, the parameter $N_1, N_2, N_3$ indicate the number of units in every hidden layer of CNNs, and it is set by experience. To find the best setting of $N_1, N_2, N_3$ for protein secondary structure prediction, we set $N_1 = N_2 = N_3 = 100$, and do several experiments with different value of $N_1, N_2, N_3$, the results on CB513 dataset can be seen in Fig. 3. Through the figure, we can see that when $N_1 = 1000$, CDNN gets the best results on CB513 dataset; So we set $N_1 = 1000$ for all datasets in other experiments in this paper.

D. EXPERIMENTS WITH DIFFERENT NUMBER OF UNITS IN LSTM

In CDNN architecture, the parameter $N_4$ indicates the number of units in hidden layer of LSTM, and it is set by experience too. To find the best setting of $N_4$ for protein secondary structure prediction, we set $N_1 = N_2 = N_3 = 100$, and do several experiments with different value of $N_4$, the results on CB513 dataset can be seen in Fig. 4. Through the figure, we can see that when $N_4 = 1000$, CDNN gets the best results on CB513 dataset; So we set $N_4 = 1000$ for all datasets in other experiments in this paper.

V. CONCLUSION

This paper proposes a novel supervised learning algorithm CDNN to address the protein secondary structure prediction problem. CDNN combines multiple CNNs and LSTM in one deep architecture to improve the discriminate ability of deep architecture. The proposed new architecture can extract amino acid features with multiple CNNs, and combine these features and raw features to train large LSTM networks. Moreover, LSTM can extract the sequence information naturally, so CDNN can extract the amino acid sequence features effectively. Our experiments conducted on two datasets demonstrate that CDNN can reach very competitive
classification performance with supervised learning. Experiments are also designed to verify the effectiveness of CDNN method with different number of hidden units in CNNs and LSTM, the results show that CDNN can reach competitive performance with proper parameters. The effect of CDNN method shows that supervised learning with deep architecture can be widely used in real applications.

REFERENCES

[1] J. Söding, “Big-data approaches to protein structure prediction,” Science, vol. 355, no. 6322, pp. 248–249, Jan. 2017.

[2] J. Zhou and O. Troyanskaya, “Deep supervised and convolutional generative stochastic network for protein secondary structure prediction,” in Proc. Int. Conf. Mach. Learn., Beijing, China, 2014, pp. 745–753.

[3] Y. D. Yang, J. Z. Gao, J. H. Wang, R. Heffernan, J. Hanson, K. Paliwal, and Y. Q. Zhou, “Sixty-five years of the long march in protein secondary structure prediction: The final stretch?” Briefings Bioinf., vol. 19, no. 3, pp. 482–494, 2018.

[4] M. Spencer, J. Eickholt, and J. Cheng, “A deep learning network approach to Ab initio protein secondary structure prediction,” IEEE/ACM Trans. Comput. Biol. Bioinf., vol. 12, no. 1, pp. 103–112, Jan. 2015.

[5] S. K. Sönderby and O. Winther, “Protein secondary structure prediction with long short term memory networks,” 2014, arXiv:1412.7828. [Online]. Available: http://arxiv.org/abs/1412.7828

[6] S. Wang, J. Peng, J. Ma, and J. Xu, “Protein secondary structure prediction using deep convolutional neural fields,” Sci. Rep., vol. 6, no. 1, pp. 1–11, May 2016.

[7] Z. Li and Y. Yu, “Protein secondary structure prediction using cascaded convolutional and recurrent neural networks,” 2016, arXiv:1604.07176. [Online]. Available: http://arxiv.org/abs/1604.07176

[8] R. Heffernan, Y. Yang, K. Paliwal, and Y. Zhou, “Capturing non-local interactions by long short-term memory bidirectional recurrent neural network for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility,” Bioinformatics, vol. 33, no. 18, pp. 2842–2849, Sep. 2017.

[9] Y. Wang, H. Mao, and Z. Yi, “Protein secondary structure prediction by using deep learning method,” Knowl.-Based Syst., vol. 118, pp. 115–123, Feb. 2017.

[10] A. Busia and N. Jaitly, “Next-step conditioned deep convolutional neural networks improve protein secondary structure prediction,” 2017, arXiv:1702.03865. [Online]. Available: http://arxiv.org/abs/1702.03865

[11] C. Fang, Y. Shang, and D. Xu, “MUFOLD-SS: New deep inception-inside-inception networks for protein secondary structure prediction,” Proteins: Struct. Function, Bioinf., vol. 86, no. 5, pp. 592–598, May 2018.

[12] R. Heffernan, K. Paliwal, J. Lyons, J. Singh, Y. Yang, and Y. Zhou, “Single-sequence-based prediction of protein secondary structures and solvent accessibility by deep whole-sequence learning,” J. Comput. Chem., vol. 39, no. 26, pp. 2210–2216, Oct. 2018.

[13] J. Hanson, K. Paliwal, T. Litfin, Y. Yang, and Y. Zhou, “Improving prediction of protein secondary structure, backbone angles, solvent accessibility and contact numbers by using predicted contact maps and an ensemble of recurrent and residual convolutional neural networks,” Bioinformatics, vol. 35, no. 14, pp. 2403–2410, Jul. 2019.

[14] M. S. Klausen, M. C. Jespersen, H. Nielsen, K. K. Jensen, V. I. Jurtz, C. K. Sönderby, M. O. A. Sommer, O. Winther, M. Nielsen, B. Petersen, and P. Marcatili, “NetSurfP-2.0: Improved prediction of protein structural features by integrated deep learning,” Proteins: Struct. Function, Bioinf., vol. 87, no. 6, pp. 520–527, 2019.

[15] S. Min, B. Lee, and S. Yoon, “Deep learning in bioinformatics,” Brief Bioinform., vol. 18, no. 5, pp. 851–869, 2016.

[16] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[17] W. Kabsch and C. Sander, “Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features,” Biopolymers, vol. 22, no. 12, pp. 2577–2637, Dec. 1983.

[18] B. Rost and C. Sander, “Improved prediction of protein secondary structure by use of sequence profiles and neural networks,” Proc. Nat. Acad. Sci. USA, vol. 90, no. 6, pp. 7558–7562, 1993.

[19] D. T. Jones, “Protein secondary structure prediction based on position-specific scoring matrices,” J. Mol. Biol., vol. 292, no. 2, pp. 195–202, 1999.

[20] S. Hua and Z. Sun, “A novel method of protein secondary structure prediction with high segment overlap measure: Support vector machine approach,” J. Mol. Biol., vol. 308, no. 2, pp. 397–407, Apr. 2001.

[21] J. J. Ward, L. J. McGuffin, B. F. Buxton, and D. T. Jones, “Secondary structure prediction with support vector machines,” Bioinformatics, vol. 19, no. 13, pp. 1650–1655, Sep. 2003.

[22] Z. Aydin, Y. Alultanbasak, and M. Borodovsky, “Protein secondary structure prediction for a single-sequence using hidden semi-markov models,” BMC Bioinf., vol. 7, no. 1, p. 178, 2006.

[23] O. Dor and Y. Zhou, “Achieving 80% ten-fold cross-validated accuracy for secondary structure prediction by large-scale training,” Proteins, Struct. Function, Bioinf., vol. 66, no. 4, pp. 838–845, 2007.

[24] X.-Q. Yao, H. Zhu, and Z.-S. She, “A dynamic Bayesian network approach to protein secondary structure prediction,” BMC Bioinf., vol. 9, no. 1, p. 49, 2008.

[25] Z. Wang, F. Zhao, J. Peng, and J. Xu, “Protein 8-class secondary structure prediction using conditional neural fields,” PROTEOMICS, vol. 11, no. 19, pp. 3786–3792, Oct. 2011.

[26] J. Dongardive and S. Abraham, “Reaching optimized parameter set: Protein secondary structure prediction using neural network,” Neurocomput., vol. 28, no. 8, pp. 1947–1974, Aug. 2017.

[27] S. Xie, Z. Li, and H. Hu, “Protein secondary structure prediction based on the fuzzy support vector machine with the hyperplane optimization,” Gene, vol. 642, pp. 74–83, Feb. 2018.

[28] M. Torrisi, M. Kalee, and G. Pollastri, “Porter 5: State-of-the-art ab initio prediction of protein secondary structure in 3 and 8 classes,” bioRxiv, Mar. 2018, Art. no. 289033.

[29] D. W. A. Buchan and D. T. Jones, “The PSIPRED protein analysis workbench: 20 years on,” Nucleic Acids Res., vol. 47, no. W1, pp. W402–W407, Jul. 2019.

[30] G. Wang and R. L. Dunbrack, “PISCES: A protein sequence culling server,” Bioinformatics, vol. 19, no. 12, pp. 1589–1591, Aug. 2003.

[31] J. A. Cuff and G. J. Barton, “Evaluation and improvement of multiple sequence methods for protein secondary structure prediction,” Proteins, Struct., Function, Genet., vol. 34, no. 4, pp. 508–519, Mar. 1999.

SHUSEN ZHOU received the Ph.D. degree in computer application technology from the Harbin Institute of Technology, in 2012. He is currently an Associate Professor with the School of Information and Electrical Engineering, Ludong University, Yantai, China. His main research interests include machine learning, pattern recognition, bioinformatics, and image processing. His current research interests include structural bioinformatics and genome.

HAILIN ZOU received the Ph.D. degree in computer application technology from the China University of Mining and Technology, Beijing, in 2005. He is currently a Full Professor with the School of Information and Electrical Engineering, Ludong University, Yantai, China. His current research interests include 3-D visualization, pattern recognition, and image processing.
CHANJUAN LIU received the Ph.D. degree in computer application technology from the China University of Mining and Technology, Beijing, in 2013. She is currently an Associate Professor with the School of Information and Electrical Engineering, Ludong University, Yantai, China. Her current research interests include machine learning, pattern recognition, and image processing.

MUJUN ZANG received the Ph.D. degree in communication and information system from Jilin University, in 2015. She is currently an Associate Professor with the School of Information and Electrical Engineering, Ludong University, Yantai, China. Her research interests include signal and image processing, pattern recognition, and machine learning methods in computer vision. Her current research interests include object detection and image classification.

TONG LIU received the Ph.D. degree in communication and information system from Jilin University, in 2016. He is currently an Associate Professor with the School of Information and Electrical Engineering, Ludong University, Yantai, China. His major research interests are machine learning, pattern recognition, and intelligent systems, including biomedical signal and image processing. His current research focus is ECG recognition.

***