2017 3rd International Symposium on Mechatronics and Industrial Informatics (ISMII 2017)
ISBN: 978-1-60595-501-8

Protein Secondary Structure Prediction Based on Deep Learning

Lin Zheng[1,a], Hong-ling Li[1,*], Nan Wu[1,b] and Li Ao[2,c]

[1]College of information, Yunnan University, Kunming

[2]College of software, Yunnan University, Kunming

[a]zhenglynne@163.com, [*]honglingli66@126.com

Keywords: Computational biology, Deep learning, Protein secondary structure

Abstract. Prediction of protein secondary structure from the amino acid sequence is a classical bioinformatics problem in computational biology. For accurate predicting the sequence-structure mapping relationship between protein secondary structure and features, a novel deep learning prediction model is proposed by combining convolutional neural network (CNN) and bi-directional recurrent neural network (BRNN) with long short-term memory cells (Bi-directional LSTM RNN). In order to draw eight classes (Q8) protein secondary structure prediction results, we first utilize CNN to filter and sample amino acid sequences, and then use Bi-directional LSTM RNN to model context information interaction between amino acids in protein. Experimental results show that the prediction accuracy of the proposed model is about 1-3% higher than that of the existing prediction models, and the prediction accuracy of 69.4% is obtained.

1. Introduction

Protein secondary structure (PSS) is developed by the protein polypeptide chain through the hydrogen bond in a certain direction coiled and folded to form a regular repetition of the conformation. In this paper, PSS was defined as eight classes (Q8) using the DSSP partitioning method developed by Sander et al [1]. Here are 3_{10}-helix (G), α-helix (H), π-helix (I), β-strand (E), β-bridge (B), β-turn (T), loop (L) and high curvature regions (S). The Q8 secondary structure is a 3-state (Q3) of the secondary structure of a refinement. PSS prediction is usually evaluated by Q3 and Q8 accuracy. Compared with the prediction of the coarse-grained Q3 PSS, the fine-grained Q8 PSS prediction can provide richer information and be more challenge [2]. At present, the prediction of the Q8 PSS has a breakthrough. In 2014, Zhou and Troyanska [3] proposed the use of a convolution of the Generative Stochastic Network (GSN) method, and then Søren Kaae Sønderby et al. [4] proposed a method based on Bi-directional LSTM, which obtained 66.4%, 67.4% Q8 accuracy respectively. In 2016, a method of the deep convolutional neural field (Deep-CNF) obtained 68.3% Q8 accuracy in [5].

In this paper, a novel deep learning prediction model based on CNN and Bi-directional LSTM RNN is presented to predict Q8 PSS. Compare with the previous methods, our model achieves better Q8 prediction accuracy.

2. Amino Acids Coding

The coding method of amino acids has an important effect on the analysis of amino acid sequences. The appropriate coding method can make the computer availably recognize the amino acid sequence and obtain satisfactory prediction results. At present, the common amino acid coding [6] often contains Orthogonal coding, 5-bit coding, Codon coding and Profile coding. The orthogonal coding known as 21-bit orthogonal coding can be used to uniquely represent a single amino acid with 20 binary numbers, and the coding vectors of the different amino acids are orthogonal to each other. Table 1 gives the orthogonal coding of amino acids, where the first 20 letters A, C, E, D, G, F, I, H, K, M, L, N, Q, P, S, R, T, W, V and Y are the basic amino acids that make up the protein chain. However, X represents the remaining amino acids, and the specific type is unknown.

Table 1. The orthogonal coding of amino acids.

Amino acids	Orthogonal coding	Amino acids	Orthogonal coding
A	10000000000000000000	L	00000000010000000000
C	01000000000000000000	N	00000000001000000000
E	00100000000000000000	Q	00000000000100000000
D	00010000000000000000	P	00000000000010000000
G	00001000000000000000	S	00000000000001000000
F	00000100000000000000	R	00000000000000100000
I	00000010000000000000	T	00000000000000010000
H	00000001000000000000	W	00000000000000001000
K	00000000100000000000	V	00000000000000000100
M	00000000010000000000	Y	00000000000000000010
X	00000000000000000001		

In addition, in order to effectively train the classifier, the position-specific scoring matrices (PSSM) spectral coding method of protein chain, which is based on the multi-sequence alignment of protein chains in the target protein chain and protein sequence data, is adopted. We combine PSSM with the orthogonal coding of amino acids to generate a 42-dimensional vector for amino acid coding, where the PSSM encoding of amino acids is a vector of 20 numerical values.

3. PSS Prediction Model

The network structure in this paper is constructed from the deep learning network models, which is described in detail below:

Neural Network. The fully connected network layer is commonly referred to be as the dense layer or multi-layer perception (MLP), also known as artificial neural network (ANN). PSS prediction is a nonlinear transformation problem. In this paper, we make use of the standard MLP for classifying protein sequence data, which is minutely defined in Figure 1:

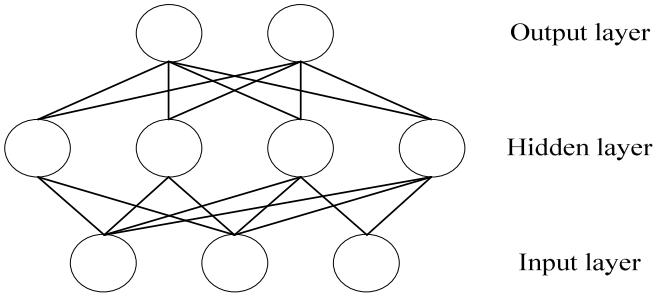

Figure 1. The single hidden layer MLP.

The linear function of the single hidden layer MLP f: $R^D \rightarrow R^L$, D is the size of the input vector x, and L is the size of the output vector $f(x)$. So the standard MLP is defined in the following linear algebraic expression:

$$f(x) = G(b^2 + W^{(2)}(s(b^1 + W^{(1)}x))) . \qquad (1)$$

Where b^1 and b^2 denote the bias vector, $W^{(1)}$ and $W^{(2)}$ denote the weight vector, G and s denote the activation function, the vectors b^1 and $W^{(1)}$ represent the bias and weight vectors of the input layer to the hidden layer, and b^2 and $W^{(2)}$ represent the bias and weight vectors of the hidden layer to the output layer. The common activation function G or s usually includes: Sigmoid, Hyperbolic Tangent function (Tanh), Rectified Linear Units (ReLU) and Leaky ReLU et al. In [8], the convergence rate of the stochastic gradient descent (SGD) obtained by the ReLU function is higher than that of the Sigmoid and Tanh, and the gradient is not saturated. However the ReLU will be "fragile" in training, which may lead to "dead" of the neurons. But the Leaky ReLU function

effectively solves the problem above. In this paper, only in the right way that if we can set up a suitable smaller learning rate can we also use the ReLU function to avoid the problem which wouldn't happen too often.

For avoiding the phenomenon of over-fitting in the training process, we adopt the dropout regularization method proposed by Nitish Srivastava et al. [9]. In the each iteration of the neural network training process, some neurons can stop working with given probability *p*. so the activation output of these neurons will be discard. Where $p_i \in [0,1]$, the dropout regularization method is defined as follows:

$$r_j^{(l)} \sim Bernoulli(p). \quad (2)$$

$$\tilde{y}^{(l)} = r^{(l)} * y^{(l)}. \quad (3)$$

$$z_i^{(l+1)} = w_i^{(l+1)} \tilde{y}^l + b_i^{(l+1)}. \quad (4)$$

$$y_i^{(l+1)} = f(z_i^{(l+1)}). \quad (5)$$

In addition, in order to improve the speed of model training and the convergence of data, we further adopt the Batch Normalization (BN) method [10], which is defined as follows,

$$\hat{x}^{(i)} = \frac{x^{(i)} - E\left[x^{(i)}\right]}{\sqrt{Var\left[x^{(i)}\right]}}. \quad (6)$$

$$U^{(i)} = \gamma^j \hat{x}^{(i)} + \beta^{(i)}. \quad (7)$$

Where *x* is the input layer vector, *E* is the mean, *Var* is the variance. The introduction of *r* and β, two learning parameters, ensure the expression of the model. *U* is the output. BN is applied to a layer after a linear change or before a nonlinear transformation. Such as, when BN method is applied to the fully connected layer in the model

$$\Phi(x) = W^1 x. \quad (8)$$

$$U(x) = batch - nom(\Phi(x)). \quad (9)$$

$$U' = s(U(x)). \quad (10)$$

It can be seen that the bias vector b^1 is shifted out, because an offset value β is applied in BN.

Convolutional Neural Network. The implementation of 1D convolutional layer is shown below:

$$z_{l+1}^{id} = x_l^i W_{l+1}^d + \beta_{l+1}^d. \quad (11)$$

$$h_{l+1}^{id} = s(z_{l+1}^{id}). \quad (12)$$

Where the input $x_l^i \in R^{kc \times 1}$ represents a co-located vector of length *k* in *c* channels and *i* means the specific co-located vector. The weight matrix *W* represents the spatial weight filter with *kc* connections between the input layer and each neuron in the output layer in the d^{th} output channel. Moreover, β is a scalar. In [5], the convolutional layers are configurable to their filter size *k* and number of filters *d*, which determine the structure of the CNN and affect the training performance of the model to a certain extent.

Bi-directional LSTM RNN. In the process of training the model, we use the bi-directional LSTM which not only overcomes the problem of gradient disappearance well but also adaptively learns and models the interaction of the context information of amino acid sequence data through its memory mechanism. Obtain the amino acid sequence information $x_1, \cdots, x_t$ before the target x_t at

time t by the forward LSTM. Moreover, the amino acid sequence information $x_1,\cdots,x_t$ after the target x_t can be obtained by the backward LSTM. Where $t=[1,2,3,\cdots,T]$, T is the sequence length. This can result in complete amino acid sequence information. In particular, the basic unit structure of the LSTM without peepholes as shown in [11].

$$i_t=\sigma(W_{xi}x_t+W_{hi}h_{t-1}+b_i). \quad (13)$$

$$f_t=\sigma(W_{xf}x_t+W_{hf}h_{t-1}+b_f). \quad (14)$$

$$o_t=\sigma(W_{xo}x_t+W_{ho}h_{t-1}+b_o) \quad (15)$$

$$g_t=\tanh(W_{xg}x_t+W_{hg}h_{t-1}+b_g) \quad (16)$$

$$c_t=f_t\odot c_{t-1}+i_t\odot g_t \quad (17)$$

$$h_t=o_t\odot\tanh(c_t) \quad (18)$$

Where $\odot$ demotes that the two vectors are multiplied by elements, h_t^{l-1} the hidden state for the current layer, c_t^{l-1} the internal memory vector, i_t the input gateway vector, f_t the forgotten gateway vector, o_t the output gateway vector, g_t the candidate node vector, c_t the internal memory vector.

4. Overall Architecture Model

The structure of Q8 PSS prediction model is showed in Figure.2, which is composed of MLP, CNN and Bi-directional LSTM RNN. The first three convolutional layers with the different filter sizes on the input all use the BN method, which all have 16 filters with receptive field sizes of 3, 5 and 7. The fourth layer with BN is a fully connected network, which have 200 hidden units. The fifth and sixth layers, the LSTM RNN, are working in the forwards and backwards direction, which have 400 hidden units. The seventh layer, a dropout layer, is set dropout rate of 0.5 and then into the eight layer known as a dense layer. The output layer providing us with a probability of PSS in a sequence uses the softmax function with output 8-class. In order to optimize the prediction model, we define the cross entropy loss function based on the probability output of the neural network and the target output sequence.

$$L(x,y)=-\sum_c y_c\ln(f_c(x)). \quad (19)$$

Where x is the input vector, y is the output vector of the true target, $f(x)$ is the neural networks probability prediction and c is the PSS class. In addition, we further define the cross entropy loss function by combining the L_2 regularization method in the following.

$$L_{reg}(x,y)=L(y,f(x))+regterm(\lambda,\theta). \quad (20)$$

Where x is a tuneable hyperparameter, and θ is the weight in the neural network.

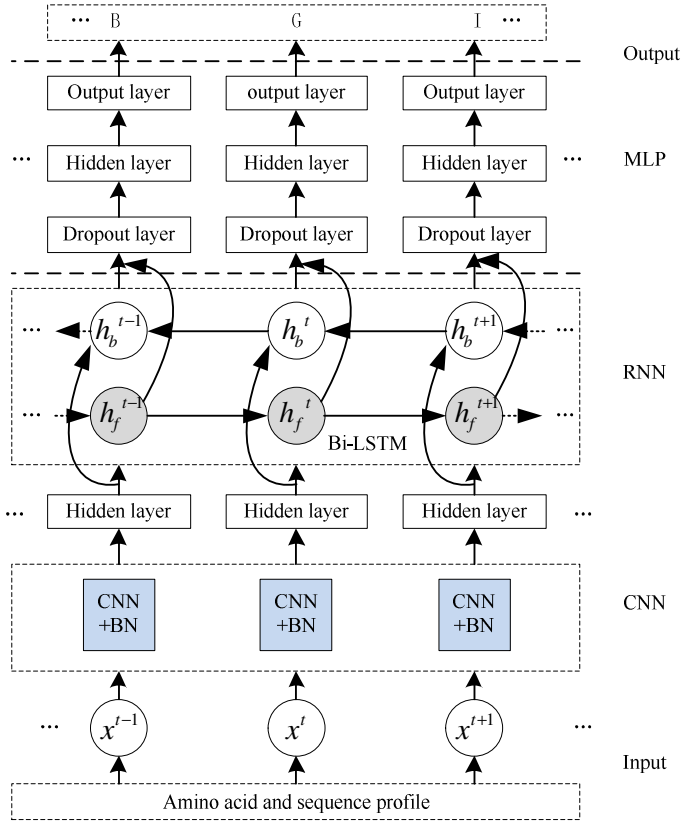

Figure 2. Protein secondary structure prediction model.

5. The Experiments

The Dataset. The Q8 PSS is predicted based on the CullPDB dataset [13] and the CB513 dataset [14]. Among them, the CullPDB dataset is a non-redundant PSS dataset containing 6128 protein amino acid sequences, each of which is labeled as a Q8 PSS [15]. To optimize the model of the super-parameters and for the purpose of testing performance on CB513 dataset, the CullPDB dataset is randomly divided into training set and verification set, which filtered to remove redundancy with CB513 dataset also known as a test set.

Experiment Results and Analysis. In order to effectively optimize the deep learning network, we use the adaptive learning rate method RMSProp proposed by Geoff Hinton [13] to train the network parameters. The impulse coefficient is set to 0.9, the learning rate is 0.005, the threshold of the normalized gradient shear is 20, and the batch size is set to 64. Stop training when the verification error rate is no longer reduced. Table 2 shows that the proposed model in this paper shows a very good performance, and access to a higher Q8 prediction accuracy based on CB513 dataset. Compared with the known previous results, our model performance of 69.4% outperforms best.

Table 2. Performance comparison with other network models.

Network models	Q8 [%]
Zhou & Troyanska,2014[3]	66.4
SØnderby et al.2014[4]	67.4
Deep-CNF.2016 [5]	68.3
This paper	**69.4**

6. Conclusion

In this paper, a prediction model based on CNN and Bi-directional LSTM CNN is proposed by predicting the Q8 PSS. In the process of establishing this model, BN method can effectively improve the optimization ability and prediction speed of the whole network model, so that the model can obtain high prediction accuracy on the CB513 dataset and is obviously better than GSN, Bi-directional LSTM CNN and deep convolution nerve field and so on.

With the study of PSS prediction based on deep learning, we find that there are some shortcomings in the establishment of the prediction model. In the future study and research, for improving the network modeling ability and the PSS prediction accuracy, it is very import to further explore and study the optimal amino acid sequence coding, the Bi-directional LSTM CNN and the SGD algorithm.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (61562090).

References

[1] W Kabsch, C Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features [J]. Biopolymers, 22(12):2577–2637, (1983).

[2] L Zhang, Li Zheng, F Zheng, W Yang. Prediction of eight-class protein secondary structure based on deep learning [J]. Journal of Computer Applications, 37(5):1512-1515, (2017).

[3] J Zhou, O G Troyanskaya. Deep supervised and convolutional generative stochastic network for protein secondary structure prediction [J]. Computer Science, 745-753, (2014).

[4] S K Sønderby, O Winther. Protein secondary structure prediction with long short term memory networks [J]. Computer Science, (2014).

[5] S Wang, J Peng, J Ma, et al. Protein secondary structure prediction using deep convolutional neural fields [J]. Scientific reports, 6, (2016).

[6] L Zhang. Research on prediction algorithm of protein secondary structure based on neural network [D]. Henan: Science of Computer, Henan University, (2016).

[7] D T Jones. Protein secondary structure prediction based on position-specific scoring matrices [J]. Journal of Molecular Biology, 292(2):195-202, (1999).

[8] K He, X Zhang, S Ren, et al. Delving Deep into Rectifiers: Surpassing human-level performance on ImageNet classification [C]//Proceedings of the IEEE international conference on computer vision.1026-1034, (2015).

[9] N Srivastava, G Hinton, A Krizhevsky, et al. Dropout: A simple way to prevent neural networks from overfitting [J]. The Journal of Machine Learning Research, 15(1): 1929-1958, (2014).

[10] S Ioffe, C Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift [J]. arXiv preprint arXiv:1502.03167, (2015).

[11] A Graves. Supervised sequence labelling with Recurrent Neural Networks. Springer, ISBN 978-3-642-24797-2, (2012).

[12] J Zhou, O G Troyanskaya. Deep supervised and convolutional generative stochastic network for protein secondary structure prediction[C]// the 31st International Conference on Computer Vision. Beijing, China:, 2014: 745-753.

[13] G Wang, Jr R L Dunbrack. PISCES: a protein sequence culling server [J]. Bioinformatics, 19(12): 1589-1591, (2003).

[14] Z Wang, F Zhao, J Peng, et al. Protein 8-class secondary structure prediction using conditional neural fields [J]. Proteomics, 11(19): 3786-3792, (2011).

[15] T Tieleman, G Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude [J]. COURSERA: Neural networks for machine learning, 4(2), (2012).