Protein-DNA Binding Residues Prediction Using a Deep Learning Model with Hierarchical Feature Extraction

Shixuan Guan, Quan Zou, Hongjie Wu*, and Yijie Ding*

Abstract—Biologically important effects occur when proteins bind to other substances, of which binding to DNA is a crucial one. Therefore, accurate identification of protein-DNA binding residues is important for further understanding of the protein-DNA interaction mechanism. Although wet-lab methods can accurately obtain the location of bound residues, it requires significant human, financial and time costs. There is thus an urgent need to develop efficient computational-based methods. Most current state-of-the-art methods are two-step approaches: the first step uses a sliding window technique to extract residue features; the second step uses each residue as an input to the model for prediction. This has a negative impact on the efficiency of prediction and ease of use. In this study, we propose a sequence-to-sequence (seq2seq) model that can input the entire protein sequence of variable length and use two modules, Transformer Encoder Block and Feature Extracting Block, for hierarchical feature extraction, where Transformer Encoder Block is used to extract global features, and then Feature Extracting Block is used to extract local features to further improve the recognition capability of the model. The comparison results on two benchmark datasets, namely PDNA-543 and PDNA-41, prove the effectiveness of our method in identifying protein-DNA binding residues. The code is available at https://github.com/ShixuanGG/DNA-protein_binding_residues.

Index Terms—Protein-DNA Binding Residues, Deep Learning, Transformer-Based Models, Hierarchical Feature Extraction.

1 Introduction

PROTEIN is a very important substance in our body that can be combined with many other substances, such as other biological macromolecules (DNA, RNA, nucleotides, etc.) or metal ions (Mn^{2+}, Zn^{2+}, Fe^{3+}, Ca^{2+}, Na^{1+}, etc.), to perform specific life activities [1], [2], [3]. The binding of proteins to DNA and thus making them interact with each other is one of the most important of these. The binding of proteins to DNA controls many DNA-related life activities, such as DNA shearing, DNA replication, and transcriptional regulation, etc [4]. Also, studying protein-DNA binding residues can help us further understand the mechanism of protein-DNA interactions [5].

Given the importance of protein-DNA binding, many wet-lab methods have emerged to recognize protein-DNA binding residues. Common wet-lab methods include: X-ray crystallography [6], Fast ChIP [7] and electrophoretic mobility shift assays (EMSAs) [8], [9]. Although accurate identification results can be obtained using these wet-lab methods, it also has some problems, such as it is costly and labor intensive, and it does not meet the growth rate of protein sequences in the post-genomic era [10]. Accordingly, it is necessary to develop an efficient and convenient computation-based method for the identification of protein-DNA binding residues. With the development of computer theory, a number of computational methods have emerged to identify protein-DNA binding residues. In general, we can divide these computational methods into three categories: sequence-based methods, structure-based methods and hybrid methods [11].

Sequence-based methods are the focus and the difficulty of research in the field of bioinformatics. Because there is less information contained in protein sequences, this leads to the disadvantage that using only sequence-based features to predict protein-DNA binding residues may have poor performance. However, the number of protein sequences is increasing day by day, and therefore, using only sequence features is the focus of research in this area. In the last decade or so, many sequence-based methods have been proposed, which contain: BindN [12], ProteDNA [13], DP-Bind [14], BindN+ [15], MetaDBSite [16], TargetDNA [17], DNABind [18], DNAPred [19] and PredDBR [20], among others. In BindN, they used three types of features extracted from protein sequences, namely hydrophobicity, side chain pK_a value and molecular mass of an amino acid, and fed them into a support vector machine (SVM) to predict protein-DNA binding residues. In DP-Bind, they used evolutionary information extracted from protein sequences, i.e., position-specific scoring matrix (PSSM) [21], and combined

This work is supported by the National Natural Science Foundation of China (61902272, 62073231, 62176175, 61876217, 61902271), National Research Project (2020YFC2006602), Provincial Key Laboratory for Computer Information Processing Technology, Soochow University (KJS2166), Opening Topic Fund of Big Data Intelligent Engineering Laboratory of Jiangsu Province(SDGC2157). (Corresponding authors : Hongjie Wu and Yijie Ding)

S. Guan and H. Wu are with the School of Electronic and Information Engineering, Suzhou University of Science and Technology, 215009, Suzhou, P.R.China. E-mail: 974948512@qq.com; hongjie.wu@qq.com.

Q. Zou is with the Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu, 610054, P.R.China. E-mail: zouquan@nclab.net

Y. Ding is with Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou, 324000, P.R.China. E-mail: wuxi_dyj@163.com

Manuscript received ; revised , .

three traditional machine learning methods, i.e., penalized logistic regression, SVM, and kernel logistic regression, to improve the recognition performance of protein-DNA binding residues. In TargerDNA, they used two protein sequence features, solvent accessibility and evolutionary information, and made use of an under-sampling technique to divide the raw data into multiple sub-datasets and applied multiple SVMs for ensemble learning to predict protein-DNA binding residues. In DNAPred, they proposed an under-sampling technique based on hyperplane distance for the data imbalance problem, after which one SVM was trained on each sub-dataset to perform prediction.

The structure-based methods use natural or predicted 3D structure information of proteins. This is because the 3D structure of a protein contains a large amount of information and the structure of a protein determines the function of the protein to some extent. Therefore, using protein structure information to predict protein-DNA binding residues often achieves better performance than sequence-based methods. Common structure-based methods include: DBD-Hunter [22], DNABINDPROT [23], DR_bind [24], PreDs [25], etc. All these methods mentioned above use only the structure information of the protein and ignore the information that may be contained in the protein sequence that may be helpful in predicting the protein-DNA binding residues. Therefore, hybrid methods combine protein sequence information and structure information to further improve prediction performance. Common hybrid methods include: TargetATP [26], COACH [27], TargetS [28], SVMPred [29] and NsitePred [30], etc. In PreDs, they generated the electrostatic potential, global and local curvature of the protein surface to predict protein-DNA binding residues from the 3D structure of the input protein. In DR_bind, the model automatically predicted protein-DNA binding residues by describing the protein structure using evolutionary, geometric, and electrostatic properties. In DNABINDPROT, they predicted protein-DNA binding residues based on a Gaussian network model of the energy localization centers in the structure. In COACH, they designed an algorithm named TM-SITE to infer binding sites from homologous structural templates and also an algorithm named S-SITE for sequence profile alignment based on evolutionary information, after which the results of both algorithms were combined using a SVM to predict protein-DNA binding residues.

With the success of deep learning in the fields of computer vision and natural language processing, there is now a large body of work that applies deep learning to the field of bioinformatics, such as transcription factor binding sites prediction [31], Bacteriocins Identification [32], and Drug-Drug Interaction Prediction [33]. In this work, we propose a new computational and sequence-based approach to predict protein-DNA binding residues efficiently and conveniently. Most of the previous work used a sliding window technique to pre-extract features for each residue. The input magnitude is one residue rather than the whole protein sequence at a time. Inspired by the work DeepCSeqSite [34], we propose an encoder-decoder model that enables the prediction of the entire protein sequence. We perform experiments on PDNA-543 and PDNA-41 datasets and compare with other existing methods, and comparison results demonstrate that our method can obtain competitive or even better prediction performance than other state-of-the-art methods. The highlights of our work are: (1) An encoder-decoder model capable of handling the entire protein sequence is proposed to enable end-to-end protein-DNA binding residue prediction. (2) Hierarchical protein residue feature extraction structure is proposed to extract not only global residue interrelationships, but also local residue interrelationships.

2 Method and Materials

2.1 Data set

In this study, we used PDNA-543 as a training set and PDNA-41 as an independent test set to validate the performance of our model. The PDNA-543 and PDNA-41 datasets were constructed by Hu et al. [17]. Hu et al. first collected 7186 DNA-binding proteins with clear annotations in the Protein Data Bank (PDB), and then used CD-hit software[35] to remove redundant sequences so that the identity of the remaining protein sequences was less than 30%, resulting in 584 sequences that met the requirements. After that, the 584 protein sequences were divided into two parts, the training set and the test set, containing 543 protein sequences and 41 protein sequences, respectively, i.e., the PDNA-543 dataset and the PDNA-41 dataset. The two datasets do not overlap and do not contain redundant sequences. On PDNA-543 we used ten-fold cross-validation to find the optimal model hyperparameters and compared the performance with other predictors. After that, we used the optimal hyperparameters found to train our model on the PDNA-543 and performed independent test on PDNA-41 to verify the generalization of our model.

The details of PDNA-543 and PDNA-41 are shown in Table 1. The PDNA-543 dataset contains 543 protein sequences that can bind to DNA, with only a few residues that can bind to DNA. And the identity of any two protein sequences in the PDNA-543 dataset is less than 30%. The PDNA-41 dataset is consistent with PDNA-543, except that there are 41 protein sequences in it.

TABLE 1: Details of the PDNA-543 and PDNA-41 datasets.

Dataset	No. Sequences[1]	No. positive[2], No. negative[3]	R_{DNABR}[4](%)
PDNA-543	543	(9549, 134995)	7.07
PDNA-41	41	(734, 14021)	5.24

[1]No. Sequences: number of protein sequences.
[2]No. positive: number of DNA binding residues.
[3]No. negative: number of non-DNA binding residues.
[4]R_{DNABR}: ratio of DNA binding residues.

2.2 Feature represention

As we all know, features of the input data determine the final performance of the model to a large extent. In this work, we used two kinds of features to represent each residue in the protein, namely the Position Specific Scoring Matrix (PSSM) and the predicted secondary structure (PSS).

2.2.1 PSSM

The PSSM contains the evolutionary information of the query protein. Previous related studies have demonstrated the positive impact of PSSM for many bioinformatic tasks [36], [37], [38]. In this study, we also utilize the PSSM to represent each residue. The PSSM features were generated using the multiple sequence alignment tool PSI-BLAST to search against Uniprot[39] database for three iterations and the E-value threshold was set to 10^{-3}. After that, a normalization formula was used to scale the values in the PSSM to the (0,1) interval in order to unify units with different features. The normalized formula for the PSSM is:

$$y = \frac{1}{1 + e^{-x}} \tag{1}$$

where x is each raw score in PSSM and y is the normalized score. Given a protein sequence of length L, the dimension of the PSSM features is $L * 20$.

2.2.2 Predicted Secondary Structure

There are three types of secondary structures of proteins, namely: coiled, α-helix and β-fold. Common secondary structure prediction tools, such as PSIPRED [5] and PSSpred [40], generate features of dimension 3 for each residue, and the range of each value is (0,1). In this work, we utilize the PSIPRED tool to generate the predicted secondary structure of the target protein. With this tool, given a protein sequence of length L, the dimension of the predicted secondary structure features is $L * 3$, and the three values represent the probability that the residue belongs to each of the three types of secondary structures, i.e., coiled, -helix and -fold, respectively.

2.3 Model

Predicting Protein-DNA binding residues is a binary classification issue. However, it is different from the traditional binary classification issue. Traditional binary classification issues, such as the prediction of DNA-binding proteins, classify the entire protein sequence. In contrast, the prediction of Protein-DNA binding residues classifies each residue in a protein sequence. Therefore, traditional methods use the sliding window technique to integrate features for each residue so that the residue is fed into the model as a sample and eventually classified for that residue. This kind of approach splits a large problem into smaller sub-binary classification problems.

In contrast, we propose an encoder-decoder model inspired by the seq2seq model [41], [42], which does not have to perform task splitting. We can input one whole protein sequence at a time, and the input protein lengths can be different. The overall framework of the model is shown in Figure 1.

2.3.1 Positional encoding

Transformer Encoder Block is included in the overall model of this study. Unlike traditional convolutional neural networks (CNN) [43] and recurrent neural networks (RNN) [44] that automatically include the positional features of the input, Transformer [45] is a purely attention-based model, which leads to a lack of positional information. Therefore, the positional encoding can help the model to better retain the features of the input information. In protein sequences, the sequential order of residues is very important. Because there are only 20 common residue types, while the overall length of the protein varies from tens to thousands. Different residue arrangements refer to different protein sequences. The addition of positional coding enables better expression of protein sequence features.

In this work, we use the same fixed position encoding as Transformer, i.e., we use sine and cosine functions of different frequencies to represent the position encoding, as follows:

$$E_{\text{pos}}\left(pos, 2i\right) = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \tag{2}$$

$$E_{pos}(pos, 2i+1) = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \tag{3}$$

where pos denotes the position of the residue in the protein sequence, i denotes the position in the residue feature dimension, and d_{model} denotes the dimension of the residue feature. In this way, it is able to obtain a representation of the position of each residue in the protein sequence with the same dimensionality as the residue features extracted. After that, the two are summed to be able to represent the position of the residues in the protein sequence.

2.3.2 Transformer Encoder Block

We use Transformer Encoder as part of our encoder. Transformer uses a network model based on a self-attention mechanism and it learns better global information. In this work, the protein sequence features, after positional encoding, will enter the Transform Encoder Block. The structure of Transform Encoder Block is shown in Figure 1. First, Embedded Sequences will enter into Multi-Head Attention. Self-Attention calculates the attention weight among all other residues including itself for each residue in the protein sequence. It is calculated as follows:

$$\text{Attention}\left(Q, K, V\right) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

where the query Q has dimension d_Q, the keyword K has the same dimension d_K, and the value V has dimension d_V (usually $d_Q = d_K = d_V$). And Multi-Head Attention is the projection of Q, K, V by h different linear transformations, and finally the different attention results are stitched together as follows:

$$\text{MultiHead}\left(Q, K, V\right) = \text{Concat}\left(\text{head}_1, \ldots, \text{head}_h\right)W^0 \tag{5}$$

$$\text{head}_i = \text{Attention}\left(QW_i^Q, KW_i^K, VW_i^V\right) \tag{6}$$

where W is a spatial mapping function and has the same dimension in this work.

After Multi-Head Attention, there is a residual connection that adds the input to the Multi-Head Attention output. In addition to Multi-Head Attention, there is usually a Multilayer Perceptron (MLP) module in the Transformer Encoder Block. The MLP module contains of two fully-connected layers and a nonlinear activation function. In general, too deep models will make the gradient disappear in training or make it difficult to propagate to the shallow part of the model during backpropagation, resulting in ineffective

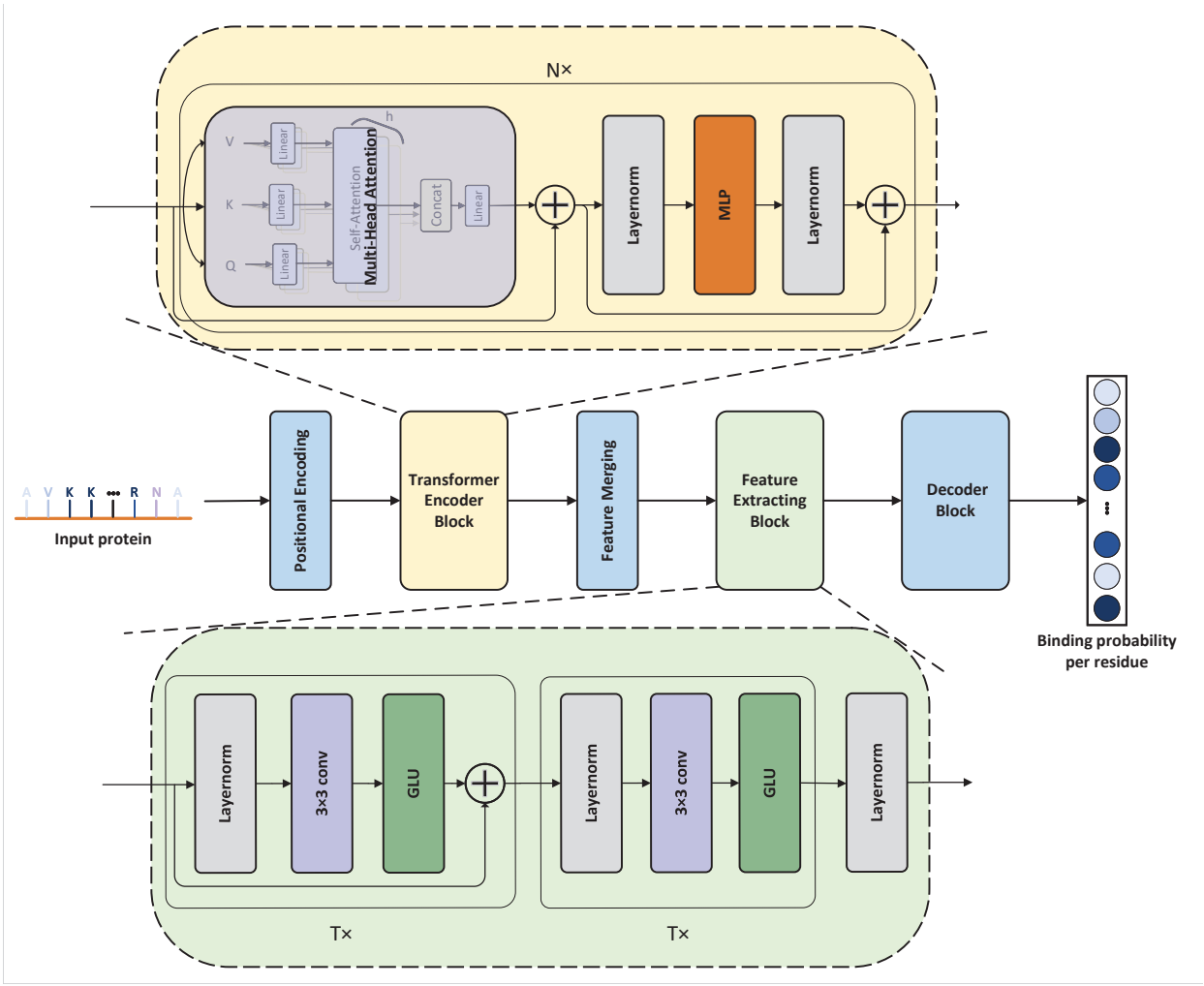

Fig. 1: Overall framework of the model.The model consists of a Positional Encoding, a hierarchical feature extraction part and a decoder part. Firstly, the protein sequence is encoded by positional encoding to encode the sequence of residues. After that, the proteins go through Transformer Encoder Block, Feature Merging and Feature Extracting Block for hierarchical feature extraction to obtain a feature representation. Finally, the Decoder Block is used to predict whether each residue can bind to DNA or not.

parameter updates. To solve this problem, so we add some residual connections after the multi-headed attention module and the MLP module.

2.3.3 Feature Extracting Block

The Feature Merging Block will integrate the features of each residue that passes through the Transformer Encoder Block. Also, after the Feature Merging Block, the dimensionality of the sequence is processed so that it can be processed by the convolution module. Specifically, the Feature Merging Block is a 2D convolutional layer. As described in the previous section, the Transformer Encoder Block is well able to extract the global features of each residue with respect to other residues. As for proteins, residues that are close in sequence tend to have similar properties. Therefore, it is also very important to learn the local features of residues. As we all know, the convolution operation is a feature aggregation of the input by sliding a convolution kernel over the feature map. The local features of residues can be well extracted by the convolution operation. Therefore, our Feature Extracting Block uses convolutional neural network for better hierarchical learning. The structure of Feature Extracting Block is shown in Figure 1.

The Feature Extracting Block is composed of 2 Layernorm-conv-GLU blocks. The main difference is that the first Layernorm-conv-GLU block is followed by the residual connection, while the second one is not. Assuming that the feature dimension is $L*1*C$ after the Feature Merging Block, the feature dimension becomes $L*1*2C$ after the Layernorm-conv. After that, the feature dimension is the same for the residual connection. Also, for the consistency of the front and back Layernorm-conv-GLU blocks, we apply the same GLU activation function in the second Layernorm-conv-GLU block. After the two Layernorm-conv-GLU blocks, the output of the encoder is obtained, which contains the global information learned from the Transformer Encoder Block and the local information learned from the CNN Extracting Block. After that, it will go to the decoder to get the final prediction result.

2.3.4 Decoder

For the decoder, the most important thing is to generate a result of the same length as the protein, which is used to determine the type of each residue (whether it is a protein-DNA binding residue or not). Here, a simple decoder based on fully connected layers that operates on residue levels can be used. Alternatively, a decoder based on multi-layer CNN can be used, but the number of output channels of the last convolutional layer is required to be 2, which is used to discriminate the type of residues. The architecture of the two types of decoders is shown in the Figure 2. In later sections, we will compare the impact on model performance using two different types of decoders. Algorithm 1 provides the pseudo-code of our method.

Algorithm 1 Pytorch-like Pseudo code for the implementation of our method.

```
# pos - Positional Encoding;
# encoder1 - Transformer Encoder Block;
# conv - Feature Merging Block;
# encoder2 - Feature Extracting Block;
# decoder - decoder based on full connection layer or CNN;

for sequence, labels in data_loader: do
# load a whole protein sequence and labels

    # adjust data dimensions and add positional encoding
    sequence_pos = pos(sequence.view(L,1,C))

    #global feature extraction via Transformer Encoder Block
    feature_global = encoder1(sequence_pos)

    #go through the Feature Merging Block for feature integration and dimension adjustment to be able to be processed further
    feature = conv(feature_global)

    # global feature extraction via Feature Extracting Block
    feature_local = encoder2(feature)

    # get predicted results by decoder
    prediction = decoder(feature_local)

    # calculate the loss and back propagate to update the model parameters
    loss = CrossEntropyLoss (prediction, labels)
    loss.backward()
    optimizer.step()
end for
```

3 RESULTS

3.1 Evaluation measurements

In this work, we use four metrics to evaluate the effectiveness of our model and its difference with other predictors, namely Matthews correlation coefficient (MCC), Specificity (SP), Sensitivity (SN), and Accuracy (ACC), which are calculated as follows:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FN)(TP+FP)(TN+FN)(TN+FP)}} \quad (7)$$

$$SP = \frac{TN}{TN+FP} \times 100 \quad (8)$$

$$SN = \frac{TP}{TP+FN} \times 100 \quad (9)$$

$$ACC = \frac{TP+TN}{TP+TN+FN+FP} \times 100 \quad (10)$$

Where TP is the predicted correct DNA residue binding site (positive sample), TN is the predicted correct non-DNA residue binding site (negative sample), FP is the incorrectly predicted non-DNA residue binding site (negative sample) as DNA residue binding site (positive sample), and FN is the incorrectly predicted DNA residue binding site (positive sample) as non-DNA residue binding site (negative sample). Larger values for all four of these metrics indicate better performance of the model.

3.2 Exploration of Optimal Model

The choice of model hyperparameters has a significant impact on the final prediction results. In this section, we use ten-fold cross-validation on the PDNA-543 dataset to compare the experimental results of different model hyperparameter choices.

3.2.1 Setting of hyperparameters based on experience

This is because a model has a large number of hyperparameters and it takes from several hours to 1 day for one experimental run. Therefore, we could not perform experimental comparisons for all hyperparameters. We empirically set some hyperparameter values, as shown in Table 2. And in subsequent experiments, the same values were used for all these hyperparameters. In particular, we set the Batch_size to 1 so that the model can handle longer protein sequences without additional padding operations.

TABLE 2: Hyperparameter values based on human experience.

Hyperparameter	Values
Optimizer	Adan
Loss_function	CrossEntropyLoss
Learning_rate	0.00005
Batch_size	1
Epoch	1000

3.2.2 Performance comparison of encoders with different number of layers

In the encoder part of our model, Transformer Encoder Blocks and Feature Extracting Blocks are used, which contain several layers of the same structure to extract features. However, using different combinations of layers, the final experimental structures produced are slightly different. Therefore, we conducted comparative experiments for different block depths to find the model hyperparameters that would give the best results. Here, the depth T of Transformer Encoder Block is taken in the range of [1,5] with a step size of 1, and the depth N of Feature Extracting Block is taken in the range of [2,10] with a step size of 2. The specific experimental settings and results are shown in Table 3. The

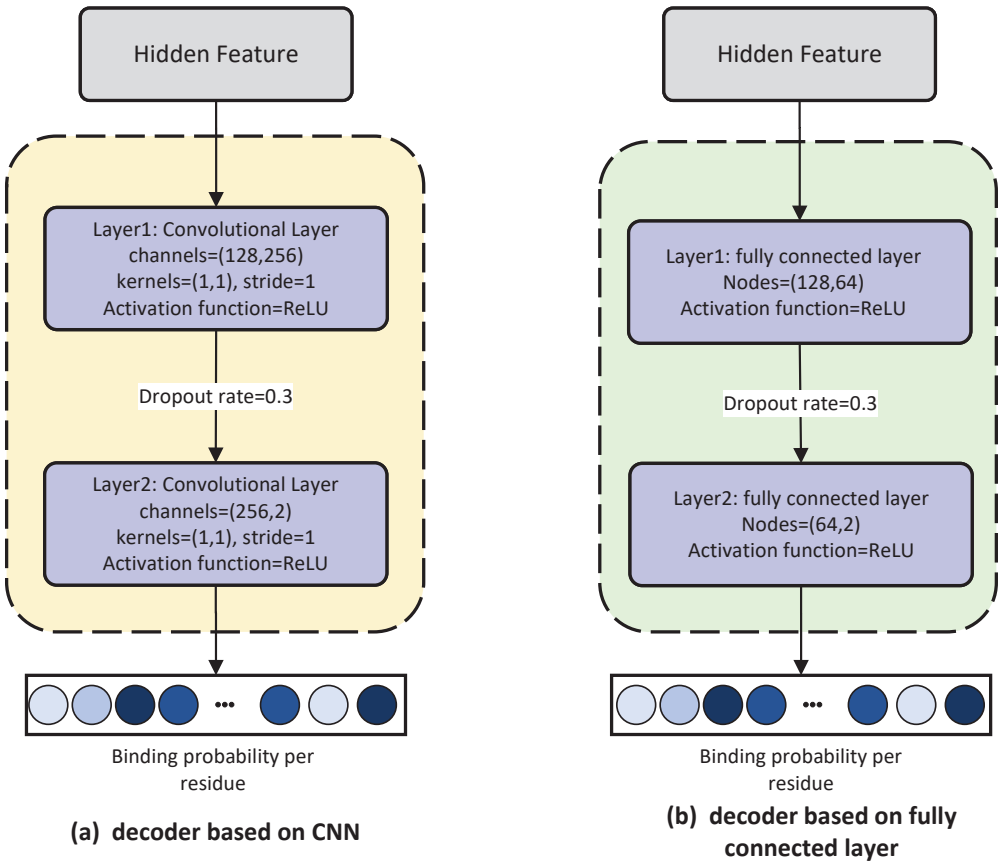

Fig. 2: The architecture of the two types of decoders.The features extracted from the encoder are passed through a CNN-based or fully-connected layer-based decoder. The CNN-based decoder consists of convolutional layers with a convolutional kernel size of 1×1 and a ReLU activation function. The fully-connected layer-based decoder consists of linear layers and a ReLU activation function. With the decoder, we are able to obtain the final prediction for each residue.

position encoding part is included in the setup of each experiment and the CNN decoder is used for the decoding part.

From Table 3 and Figure 3, we can see that the effect gets progressively better as the number of layers in the Transformer Encoder Block(T) increases. But when the number of layers in the Transformer Encoder Block reaches 3, the effect improvement is less obvious. And when T = 5, the effect on the model has reached saturation, that is, increasing T does not make the performance any better. Also, the performance of the model becomes better when the number of layers in the Feature Extracting Block(N) increases. But when the number of layers in the Feature Extracting Block(N) is greater than 6, it has an inverse effect on the performance of the model. Therefore, better experimental results are often achieved when the sum of T and N is at a more intermediate value. Here, we take $T = 5$, $N = 6$, when the model works best ($MCC = 0.3487$, $SP = 95.28\%$, $SN = 46.48\%$, and $ACC = 93.02\%$).

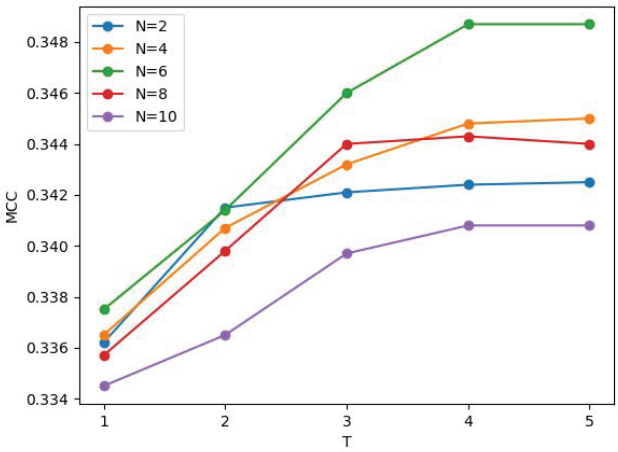

Fig. 3: MCC comparison of encoders with different number of layers.

TABLE 3: Performance comparison of encoders with different number of layers.

The number of layers in the Feature Extracting Block (N)	The number of layers in the Transformer Encoder Block (T)	MCC	SP	SN	ACC
2	1	0.3362	95.23	44.16	92.79
	2	0.3415	95.20	46.03	92.98
	3	0.3421	94.97	47.83	92.96
	4	0.3424	95.28	44.64	92.84
	5	0.3425	94.96	48.07	92.98
4	1	0.3365	95.32	42.63	92.58
	2	0.3407	95.22	45.41	92.94
	3	0.3432	95.33	44.01	92.75
	4	0.3448	95.32	44.38	92.80
	5	0.3450	95.19	47.50	93.11
6	1	0.3375	95.30	43.38	92.67
	2	0.3414	95.32	44.00	92.70
	3	0.3460	95.30	45.21	92.86
	4	0.3487	95.25	43.70	92.46
	5	0.3487	95.28	46.48	93.02
8	1	0.3357	94.92	47.43	92.93
	2	0.3398	95.23	45.08	92.00
	3	0.3440	95.30	44.67	92.79
	4	0.3443	95.23	46.21	92.97
	5	0.3440	95.27	45.17	92.86
10	1	0.3345	95.23	43.95	92.78
	2	0.3365	95.22	44.58	92.83
	3	0.3397	95.18	46.10	92.99
	4	0.3408	95.25	44.86	92.84
	5	0.3408	95.16	47.03	93.06

3.2.3 Performance comparison with and without positional coding

As previously described, the position of residues in a protein sequence is very important. This is because different combinations of residue arrangements represent different protein sequences. In this section, we explore the effect of the presence or absence of positional coding on the experimental results. The experimental comparison results are shown in Table 4. From Table 4, we can see that the effect of the model with positional coding improves in MCC, SP, and SN compared to the effect of the model without positional coding by 1.67%, 0.07%, and 0.39%, respectively, which indicates the effectiveness of positional coding to some extent. It is also evident from another aspect that although positional coding is helpful, our model still performs better without their presence.

TABLE 4: Performance comparison with and without positional coding.

With or without positional coding	MCC	SP	SN	ACC
with	0.3487	95.28	46.48	93.02
without	0.3430	95.21	46.30	93.02

Here, we set $T = 5$, $N = 6$, and the decoder uses CNN decoder.

3.2.4 Performance comparison of different decoder types

After the protein features are extracted by the encoder, the extracted features are fed into the decoder to get the final prediction results. In this section, we explore the effects of different types of decoders on the experimental results. Here we compare and analyze the experimental results of using fully connected layer and using convolutional network as decoder. The experimental results are shown in Table 5. From Table 5, we can see that the MCC, SP, SN and ACC values reach 0.3522,95.38%, 45.20% and 92.83% respectively when using fully connected layer as the decoder. Compared with using the convolutional layer as the decoder, the MCC and SP values increase significantly, while the SN and ACC values decrease slightly. Therefore, we can see that the two types of decoders have their respective strengths. Since the use of fully connected layer is simple, we uniformly use fully connected layer as the decoder of the model in the later experiments.

TABLE 5: Performance comparison of different decoder types.

Decoder Type	MCC	SP	SN	ACC
Fully-connected layer	0.3522	95.38	45.20	92.83
CNN	0.3487	95.28	46.48	93.02

Here, we set $T = 5$, $N = 6$, and with positional encoding.

3.2.5 Performance comparison of decoders with different number of layers

After selecting the decoder type, in this section, we explore the effect of different decoder layers on the model performance. We selected one, two or three fully connected layers as decoders and performed a five-fold cross-validation on the PDNA-543 dataset. In Table 6, we compare the experimental results for different decoder layers. From Table 6, we can see that the model achieves optimal performance on MCC and SP with 0.3522 and 95.38% respectively when two-layer decoder is selected. Therefore, in this work, we use two fully connected layers as hyperparameters for the final decoder selection.

TABLE 6: Performance comparison of decoders with different number of layers.

Decoder Layers	MCC	SP	SN	ACC
One-layer	0.3471	95.21	47.41	93.07
Two-layer	0.3522	95.38	45.20	92.83
Three-layer	0.3149	85.78	41.46	83.61

Here, we set $T = 5$, $N = 6$, and with positional encoding.

3.3 Performance comparison with other predictors

To more accurately evaluate the predictive performance of our proposed classifier and its robustness, we compare it with other existing methods by performing ten-fold cross-validation and independent tests on the PDNA-543 and PDNA-41 datasets, respectively.

3.3.1 Ten-fold cross-validation performance comparison on the PDNA-543 dataset

We perform a ten-fold cross-validation on the PDNA-543 dataset and take the average of each time on the validation set as the final prediction performance result. Table 7 shows our comparison with other existing method TargetDNA[17].

Due to the limitation of the model, we could not will adjust the threshold value of the classification significantly. With the default threshold, i.e., threshold = 0.5, we can reach 0.3522,95.38%, 45.20%, and 92.83% for MCC, SP, SN, and ACC, respectively, in a ten-fold cross-validation on PDNA-543 dataset. In comparison with TargetDNA, when its threshold is set to $SN \approx SP$, its MCC, SP, SN, and ACC are 0.304, 77.05%, 76.98%, and 77.04%, respectively. Since the two methods use different classification thresholds, we only compare their MCCs. it can be found that the MCC of our method (0.3522) is 15.9% higher than that of TargetDNA (0.3040). And with the threshold setting of $SP \approx 95\%$, the MCC, SP, SN and ACC of TargetDNA were 0.3390,95.00%, 40.60% and 91.40%, respectively. It can be clearly seen that our method outperforms TargetDNA in MCC, SN and ACC by 3.9%, 11.3% and 1.6%, respectively.

TABLE 7: Performance comparison of different classifiers on PDNA-543 dataset via ten-fold cross-validation.

Method	MCC	SP	SN	ACC
TargerDNA ($SN \approx SP$)	0.304	77.05	76.98	77.04
TargetDNA ($SP \approx 95\%$)	0.339	95.00	40.60	91.40
Our method	0.352	95.38	45.20	92.83

3.3.2 Independent test performance comparison on the PDNA-41 dataset

We performed independent tests on PDNA-41 to validate the robustness of our method. Table 8 shows how we compare with other existing methods containing ProteDNA[13], MataDBSite[16], BindN[12], COACH[27], DP-Bind[14], DNABind[18], BindN+[15], TargerDNA[17] and PredDBR[20]. From Table 8, we can clearly see that our method obtained satisfactory experimental results compared to the previous method. Specifically, the MCC, SP, SN and ACC values reached 0.357,96.44%, 47.57% and 94.87%, respectively. Compared to COACH, our method achieved better experimental results for all evaluation metrics including MCC, SP, SN, and ACC when tested independently on the PDNA-41 data set. Among them, MCC, SP, SN, and ACC improved by 1.4%, 1.4%, 3.0%, and 2.4%, respectively. In addition, our method outperforms other methods that adjust thresholds. For example, the MCC values of BindN+ and TargerDNA are inferior to our method, regardless of how they adjust the threshold. For the state-of-the-art method PredDBR, it achieves 0.359, 96.79%, 39.10%, and 93.93% for MCC, SP, SN, and ACC, respectively. Although PredDBR has higher MCC value (0.359) and SP (96.79%), it only improves 0.56% and 0.36%, respectively, compared with our method, which is not much different from the performance of our model. However, the performance of our model is improved by 21.7% and 1.0% on SN and ACC, respectively.

TABLE 8: Performance comparison of different classifiers on PDNA-41 dataset via independent test.

Method	MCC	SP	SN	ACC
ProteDNA	0.160	99.84	4.77	95.11
MataDBSite	0.221	93.35	34.20	90.41
BindN	0.143	80.90	45.64	79.15
COACH	0.352	95.10	46.19	92.67
DP-Bind	0.241	82.43	61.72	81.40
DNABind	0.264	80.28	70.16	79.78
BindN+ ($SP \approx 95\%$)	0.178	95.11	24.11	91.58
BindN+ ($SP \approx 85\%$)	0.213	85.41	50.81	83.69
TargerDNA ($SN \approx SP$)	0.269	85.79	60.22	84.52
TargerDNA ($SP \approx 95\%$)	0.300	93.27	45.50	90.89
PredDBR	0.359	96.79	39.10	93.93
Our method	0.357	96.44	47.57	94.87

4 Conclusion

In this study, we propose an encoder-decoder model to predict protein-DNA binding sites. To represent a protein sequence, we use two sequence-based features, the evolutionary feature PSSM and the predicted secondary structure, respectively. The main advantage of our approach is the hierarchical feature extraction of residues in protein sequences, both global and local feature representation learning. Unlike current state-of-the-art methods, our model enables end-to-end prediction of an entire protein sequence without the need for feature pre-extraction for each residue using a sliding window technique, which demonstrates the ease of use of our model. Comparing with previous methods, our model achieves respectable performance on the PDNA-41 test set (MCC:0.357, SP:96.44%, SN:47.57%, ACC:94.87%), which proves the effectiveness of our model.

While our method has made some progress and can handle variable length protein sequences, it also limits our model to one protein input at a time. Therefore, we will further try more models for the problem of inconsistent protein sequence lengths. Given the success of graph neural networks in bioinformatics [46], we will try to employ graph structures to represent protein sequences to identify DNA binding residues. In addition, the features used in this work could be improved. With the great achievements in the field of protein structure prediction in recent years, we can use the predicted structural information to aid in this task.

AVAILABILITY OF DATA AND MATERIAL

The datasets, codes and corresponding results are available at https://github.com/ShixuanGG/DNA-protein_binding_residues.

REFERENCES

[1] C. M. Dobson *et al.*, "Chemical space and biology," *Nature*, vol. 432, no. 7019, pp. 824–828, 2004.

[2] M. Gao and J. Skolnick, "The distribution of ligand-binding pockets around protein-protein interfaces suggests a general mechanism for pocket formation," *Proceedings of the National Academy of Sciences*, vol. 109, no. 10, pp. 3784–3789, 2012.

[3] J. Zhao, Y. Cao, and L. Zhang, "Exploring the computational methods for protein-ligand binding site prediction," *Computational and structural biotechnology journal*, vol. 18, pp. 417–426, 2020.

[4] Y. Ofran, V. Mysore, and B. Rost, "Prediction of dna-binding residues from sequence," *Bioinformatics*, vol. 23, no. 13, pp. i347–i353, 2007.

[5] S. Jones, P. Van Heyningen, H. M. Berman, and J. M. Thornton, "Protein-dna interactions: a structural analysis," *Journal of molecular biology*, vol. 287, no. 5, pp. 877–896, 1999.

[6] M. Smyth and J. Martin, "x ray crystallography," *Molecular Pathology*, vol. 53, no. 1, p. 8, 2000.

[7] J. D. Nelson, O. Denisenko, and K. Bomsztyk, "Protocol for the fast chromatin immunoprecipitation (chip) method," *Nature protocols*, vol. 1, no. 1, pp. 179–185, 2006.

[8] M. A. Heffler, R. D. Walters, and J. F. Kugel, "Using electrophoretic mobility shift assays to measure equilibrium dissociation constants: Gal4-p53 binding dna as a model system," *Biochemistry and Molecular Biology Education*, vol. 40, no. 6, pp. 383–387, 2012.

[9] L. M. Hellman and M. G. Fried, "Electrophoretic mobility shift assay (emsa) for detecting protein–nucleic acid interactions," *Nature protocols*, vol. 2, no. 8, pp. 1849–1861, 2007.

[10] S. Vajda and F. Guarnieri, "Characterization of protein-ligand interaction sites using experimental and computational methods." *Current opinion in drug discovery & development*, vol. 9, no. 3, pp. 354–362, 2006.

[11] Y. Ding, C. Yang, J. Tang, and F. Guo, "Identification of protein-nucleotide binding residues via graph regularized k-local hyperplane distance nearest neighbor model," *Applied Intelligence*, pp. 1–15, 2021.

[12] L. Wang and S. J. Brown, "Bindn: a web-based tool for efficient prediction of dna and rna binding sites in amino acid sequences," *Nucleic acids research*, vol. 34, no. suppl_2, pp. W243–W248, 2006.

[13] W.-Y. Chu, Y.-F. Huang, C.-C. Huang, Y.-S. Cheng, C.-K. Huang, and Y.-J. Oyang, "Protedna: a sequence-based predictor of sequence-specific dna-binding residues in transcription factors," *Nucleic acids research*, vol. 37, no. suppl_2, pp. W396–W401, 2009.

[14] S. Hwang, Z. Gou, and I. B. Kuznetsov, "Dp-bind: a web server for sequence-based prediction of dna-binding residues in dna-binding proteins," *Bioinformatics*, vol. 23, no. 5, pp. 634–636, 2007.

[15] L. Wang, C. Huang, M. Q. Yang, and J. Y. Yang, "Bindn+ for accurate prediction of dna and rna-binding residues from protein sequence features," *BMC Systems Biology*, vol. 4, no. 1, pp. 1–9, 2010.

[16] J. Si, Z. Zhang, B. Lin, M. Schroeder, and B. Huang, "Metadbsite: a meta approach to improve protein dna-binding sites prediction," *BMC systems biology*, vol. 5, no. 1, pp. 1–7, 2011.

[17] J. Hu, Y. Li, M. Zhang, X. Yang, H.-B. Shen, and D.-J. Yu, "Predicting protein-dna binding residues by weightedly combining sequence-based features and boosting multiple svms," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 14, no. 6, pp. 1389–1398, 2016.

[18] R. Liu and J. Hu, "Dnabind: A hybrid algorithm for structure-based prediction of dna-binding residues by combining machine learning-and template-based approaches," *Proteins: Structure, Function, and Bioinformatics*, vol. 81, no. 11, pp. 1885–1899, 2013.

[19] Y.-H. Zhu, J. Hu, X.-N. Song, and D.-J. Yu, "Dnapred: accurate identification of dna-binding sites from protein sequence by ensembled hyperplane-distance-based support vector machines," *Journal of chemical information and modeling*, vol. 59, no. 6, pp. 3057–3071, 2019.

[20] J. Hu, Y.-S. Bai, L.-L. Zheng, N.-X. Jia, D.-J. Yu, and G. Zhang, "Protein-dna binding residue prediction via bagging strategy and sequence-based cube-format feature," *IEEE/ACM transactions on computational biology and bioinformatics*, pp. 1–1, 2021.

[21] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped blast and psi-blast: a new generation of protein database search programs," *Nucleic acids research*, vol. 25, no. 17, pp. 3389–3402, 1997.

[22] M. Gao and J. Skolnick, "Dbd-hunter: a knowledge-based method for the prediction of dna–protein interactions," *Nucleic acids research*, vol. 36, no. 12, pp. 3978–3992, 2008.

[23] P. Ozbek, S. Soner, B. Erman, and T. Haliloglu, "Dnabindprot: fluctuation-based predictor of dna-binding residues within a network of interacting residues," *Nucleic acids research*, vol. 38, no. suppl_2, pp. W417–W423, 2010.

[24] Y. C. Chen, J. D. Wright, and C. Lim, "Dr_bind: a web server for predicting dna-binding residues from the protein structure based on electrostatics, evolution and geometry," *Nucleic acids research*, vol. 40, no. W1, pp. W249–W256, 2012.

[25] Y. Tsuchiya, K. Kinoshita, and H. Nakamura, "Preds: a server for predicting dsdna-binding site on protein molecular surfaces," *Bioinformatics*, vol. 21, no. 8, pp. 1721–1723, 2005.

[26] D.-J. Yu, J. Hu, Z.-M. Tang, H.-B. Shen, J. Yang, and J.-Y. Yang, "Improving protein-atp binding residues prediction by boosting svms with random under-sampling," *Neurocomputing*, vol. 104, pp. 180–190, 2013.

[27] J. Yang, A. Roy, and Y. Zhang, "Protein–ligand binding site recognition using complementary binding-specific substructure comparison and sequence profile alignment," *Bioinformatics*, vol. 29, no. 20, pp. 2588–2595, 2013.

[28] D.-J. Yu, J. Hu, J. Yang, H.-B. Shen, J. Tang, and J.-Y. Yang, "Designing template-free predictor for targeting protein-ligand binding sites with classifier ensemble and spatial clustering," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 10, no. 4, pp. 994–1008, 2013.

[29] K. Chen, M. J. Mizianty, and L. Kurgan, "Atpsite: sequence-based prediction of atp-binding residues," in *Proteome Science*, vol. 9, no. 1. BioMed Central, 2011, pp. 1–8.

[30] ——, "Prediction and analysis of nucleotide-binding residues using sequence and sequence-derived structural descriptors," *Bioinformatics*, vol. 28, no. 3, pp. 331–341, 2012.

[31] Q. Zhang, S. Wang, Z. Chen, Y. He, Q. Liu, and D.-S. Huang, "Locating transcription factor binding sites by fully convolutional neural network," *Briefings in Bioinformatics*, vol. 22, no. 5, 01 2021, bbaa435. [Online]. Available: https://doi.org/10.1093/bib/bbaa435

[32] Z. Cui, Z.-H. Chen, Q. Zhang, V. V. Gribova, V. F. Filaretov, and D.-s. Huang, "Rmscnn: A random multi-scale convolutional neural network for marine microbial bacteriocins identification," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pp. 1–1, 2021.

[33] X. Su, Z.-H. You, D.-s. Huang, L. Wang, L. Wong, B. Ji, and B. Zhao, "Biomedical knowledge graph embedding with capsule network for multi-label drug-drug interaction prediction," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2022.

[34] Y. Cui, Q. Dong, D. Hong, and X. Wang, "Predicting protein-ligand binding residues with deep convolutional neural networks," *BMC bioinformatics*, vol. 20, no. 1, pp. 1–12, 2019.

[35] W. Li and A. Godzik, "Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences," *Bioinformatics*, vol. 22, no. 13, pp. 1658–1659, 05 2006. [Online]. Available: https://doi.org/10.1093/bioinformatics/btl158

[36] Y. Wang, Y. Ding, F. Guo, L. Wei, and J. Tang, "Improved detection of dna-binding proteins via compression technology on pssm information," *PloS one*, vol. 12, no. 9, p. e0185587, 2017.

[37] Y. Ding, J. Tang, and F. Guo, "Identification of protein–ligand binding sites by sequence information and ensemble classifier," *Journal of Chemical Information and Modeling*, vol. 57, no. 12, pp. 3149–3161, 2017.

[38] S. Ahmad and A. Sarai, "Pssm-based prediction of dna binding sites in proteins," *BMC bioinformatics*, vol. 6, no. 1, pp. 1–6, 2005.

[39] T. U. Consortium, "UniProt: a worldwide hub of protein knowledge," *Nucleic Acids Research*, vol. 47, no. D1, pp. D506–D515, 11 2018. [Online]. Available: https://doi.org/10.1093/nar/gky1049

[40] R. Yan, D. Xu, J. Yang, S. Walker, and Y. Zhang, "A comparative assessment and analysis of 20 representative sequence alignment

methods for protein structure prediction," *Scientific reports*, vol. 3, no. 1, pp. 1–9, 2013.
[41] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
[42] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.
[43] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.
[44] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019*, 2015.
[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
[46] H.-C. Yi, Z.-H. You, D.-S. Huang, and C. K. Kwoh, "Graph representation learning in bioinformatics: trends, methods and applications," *Briefings in Bioinformatics*, vol. 23, no. 1, 09 2021, bbab340. [Online]. Available: https://doi.org/10.1093/bib/bbab340

Shixuan Guan is currently a graduate student at the School of Electronic and Information Engineering, Suzhou University of Science and Technology, Suzhou, China. His research interests include bioinformatics and deep learning. He is currently engaged in the research of protein-DNA binding residues.

Quan Zou is a professor of Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China. He received his PH.D. from Harbin Institute of Technology, P.R.China in 2009, and worked at Xiamen University and Tianjin University from 2009 to 2018. His research is in the areas of bioinformatics, machine learning and parallel computing. Now he is putting the focus on protein classification, genome assembly, annotation and functional analysis from the next generation sequencing data with parallel computing methods. Several related works have been published by Briefings in Bioinformatics, Bioinformatics, PLOS Computational Biology and IEEE/ACM TCBB.

Yijie Ding received his Ph.D. degree from the School of Computer Science and Technology at Tianjin University in 2018. Currently, he is an associate professor in the Yangtze Delta Region Institute, University of Electronic Science and Technology of China. His research interests include bioinformatics and machine learning.

Hongjie Wu received the Ph.D. degree in computer science from Soochow University in 2013. He is currently a professor at the School of Electronic and Information Engineering, Suzhou University of Science and Technology, Suzhou, China. His primary research interests include machine learning, parallel programming, protein structure and function prediction and gene expression networks. He is currently working on machine learning, membrane protein and DNA binding protein related predictions.