# DeepTF: Accurate Prediction of Transcription Factor Binding Sites by Combining Multi-Scale Convolution and Long Short-Term Memory Neural Network

Xiao-Rong Bao, Yi-Heng Zhu, and Dong-Jun Yu*

School of Computer Science and Engineering, Nanjing University of Science and Technology, 200 Xiaolingwei, Nanjing, China, 210094

**Abstract.** Transcription factor binding site (TFBS), one of the DNA-protein binding sites, plays important roles in understanding regulation of gene expression and drug design. Recently, deep-learning based methods have been widely used in the prediction of TFBS. In this work, we propose a novel deep-learning model, called Combination of Multi-Scale Convolutional Network and Long Short-Term Memory Network (MCNN-LSTM), which utilizes multi-scale convolution for feature processing, and the long short-term memory network to recognize TFBS in DNA sequences. Moreover, we design a new encoding method, called multi-nucleotide one-hot (MNOH), which considers the correlation between nucleotides in adjacent positions, to further improve the prediction performance of TFBS. Stringent cross-validation and independent tests on benchmark datasets demonstrated the efficacy of MNOH and MCNN-LSTM. Based on the proposed methods, we further implement a new TFBS predictor, called DeepTF. The computational experimental results show that our predictor outperformed several existing TFBS predictors.

**Keywords:** Transcription Factor Binding Site · Multi-Scale Convolution · Long Short-Term Memory Network · Multi-Nucleotide One-Hot.

## 1 Introduction

Accurate prediction of transcription factor binding site (TFBS) from DNA sequences is critical for regulation of gene expression and drug design [1]. Traditionally, researchers identified TFBS through biochemical methods, such as ChIP-seq [2] and ChIP-chip [3]. However, these methods are time-consuming and laborious, that cannot keep up with relevant research advances in the postgenomic era. Hence, many intelligent computational methods, such as traditional-machine-learning-based ones and deep-learning-based ones, have been proposed and achieved outstanding performance in the task of predicting TFBS [4].

* Corresponding author: Dong-Jun Yu (njyudj@njust.edu.cn)

Traditional-machine-learning-based methods [5–7] led to trends in the field of early TFBS prediction. Wong *et al.* [6] proposed kmerHMM, which used Hidden Markov Models (HMMs) to derive precise TFBS. Fletez-Brant *et al.* [5] developed kmer-SVM, which used a support vector machine with k-mer sequence features to identify predictive combinations of short TFBS. Nevertheless, there are more and more datasets related to transcription factors in the postgenomic era, and traditional-machine-learning-based methods have been unable to quickly and effectively predict TFBS in DNA sequences due to that they are designed for small-scale dataset [8].

Recently, deep-learning-based models [9, 10] have achieved good performances in prediction of TFBS. For example, Alipanahi *et al.* [9] implemented DeepBind, ascertaining sequence specificities from experimental data with Convolutional Neural Network (CNN) [11], to identify TFBS in DNA sequences. Next, CNN-Zeng was presented by Zeng *et al.*[10], which adopted CNN to capture the sequence-based features which are critical to accurate characterization of TFBS.

Although these methods have achieved remarkable performance in prediction of TFBS [12], there is still room for further enhancing the performances as the following two points. Firstly, these methods only take consideration of the independent relationship among nucleotides in TFBS and ignore the interdependence of nucleotides in the sequence. Taking DeepBind [9] as an example, which used one-hot to represent DNA sequences, it ignores the relationship between adjacent nucleotides. However, studies [13] have shown that considering the correlation between nucleotides in adjacent positions can effectively promote the prediction performance. Secondly, many CNN-based models employ a fixed-size convolution kernel to identify TFBS in DNA sequences. For example, CNN-Zeng take the fixed kernel size 24 when do prediction. However, it has a limitation that the length of the TFBS in the model is fixed. It is well known that different kinds of TFBS have different binding lengths [14]. Therefore, it will be useful to employ multiple size of convolutional kernels to capture multi-scale features.

In this work, we first propose a new encoding method, multi-nucleotide one-hot (MNOH), which takes account of the correlation between nucleotides in adjacent positions. Moreover, we design a new deep learning architecture, called Combination of Multi-Scale Convolutional Network and Long Short-Term Memory Network (MCNN-LSTM), which uses multi-scale convolution layers for feature processing, and then uses Long Short-Term Memory Network (LSTM)[15] as a recurrent model that recognizes TFBS in DNA sequences [16]. Based on above, we further implement a TFBS predictor, called DeepTF. Stringent cross-validation and independent tests on benchmark datasets have demonstrated the efficacy of our proposed methods.

## 2 Materials and Methods

### 2.1 Benchmark Datasets

We used 690 ChIP-seq datasets, constructed in CNN-Zeng [10] as benchmark datasets to evaluate the proposed method. The positive set consisted of the cen-

tering 101 bps that overlaps one TFBS, which have more than 40 000 binding events, while the negative set was 101bp sequences, which do not bind to transcription factor. For each dataset, we use 70% for training, 10% for validating and the remaining 20% for testing.

### 2.2 Feature Representation

We use the MNOH encoding method to convert DNA sequences into feature matrixes. Specifically, for one DNA sequence $S = (s_1, s_2, ...s_n)$ with the length of $n$ , we have $s_i \in A(adenine), C(cytosine), G(guanine), T(thymine)$, which means $s_i$ can be any one of the 4 nucleotide bases in DNA. MNOH encodes it as a matrix, denoted as $Z_{MNOH}$, as follows:

$$\begin{aligned} Z_{MNOH} &= [Z_1 \ Z_2 \ \cdots \ Z_{n-h+1}] \\ &= \begin{bmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,4^h} \\ z_{2,1} & z_{2,2} & \cdots & z_{2,4^h} \\ \vdots & \vdots & & \vdots \\ z_{n-h+1,1} & z_{n-h+1,2} & \cdots & z_{n-h+1,4^h} \end{bmatrix} \end{aligned} \tag{1}$$

Where $h$ represents the degree of MNOH, $Z_i$ represents an one-hot vector of the $i$-th $(i = 1, 2, ..., n - h + 1)$ nucleotide in DNA sequence, and $z_{i,j}$ is the value of $j$-th position in vector $Z_i$, which is defined as follows:

$$z_{i,j} = \begin{cases} 1 & when\ j = 4^{h-1} f(s_i) + 4^{h-2} f(s_{i+1}) + ... + 4^0 f(s_{i+h-1}) \\ 0 & others \end{cases} \tag{2}$$

Where $f(s_i)$ represents the function corresponding to the nucleotide at the $i$-th position in the DNA sequence. The specific definition of $f(s_i)$ is in eq. (3):

$$f(s_i) = \begin{cases} 1 & when\ s_i = A \\ 2 & when\ s_i = C \\ 3 & when\ s_i = G \\ 4 & when\ s_i = T \end{cases} \tag{3}$$

Finally, each DNA sequence can be encoded into a $(n - h + 1) \times 4^h$ matrix. The one-hot encoding method is a special kind of the MNOH when $h = 1$ . That means we only consider each individual nucleotide, and each nucleotide in the DNA sequence is denoted as one of the 4 one-hot vectors, *i.e.*, [1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0] and [0, 0, 0, 1]. Furthermore, when $h = 2$, MNOH takes the dependencies between two adjacent nucleotides into account, which has 16 dinucleotides totally, *i.e.*, [AA AC AG AT CA CC CG CT GA GC GG GT TA TC TG TT]. As shown in eq. (2), these dinucleotides can be denoted as one of the 16 one-hot vectors, *i.e.*, [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],..., and [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1].

However, as the degree of MNOH increases, the dimension of the vector will rise at the same time. Hence, the GPU memories consumed by the model will increase and the calculations will drop. Due to the limitation of GPU memory, we only evaluate the performances of MNOH with degree less than five.

### 2.3 Model Architecture

In this section, we introduce a novel deep learning model, named MCNN-LSTM, by combining multi-scale CNN with LSTM. Fig. 1 illustrates the workflow of MCNN-LSTM, which consists of three parts, *i.e.*, multi-scale convolutional layer, LSTM network layer and fully-connected (fc) layer. Firstly, the input DNA sequence is encoded into a $(n-h+1)\times 4^h$ matrix, by MNOH. Then, this matrix is passed on to the multi-scale convolutional layer, and each sublayer scans the output of the previous sublayer through convolutional kernels of different sizes. In other words, this layer can be thought of as feature extractors, which can capture the features of TFBS of different lengths. Next, we use LSTM as a recurrent model that recognizes TFBS in DNA sequences. At last, the fc layer takes the outputs of the LSTM network and produces two classification results.

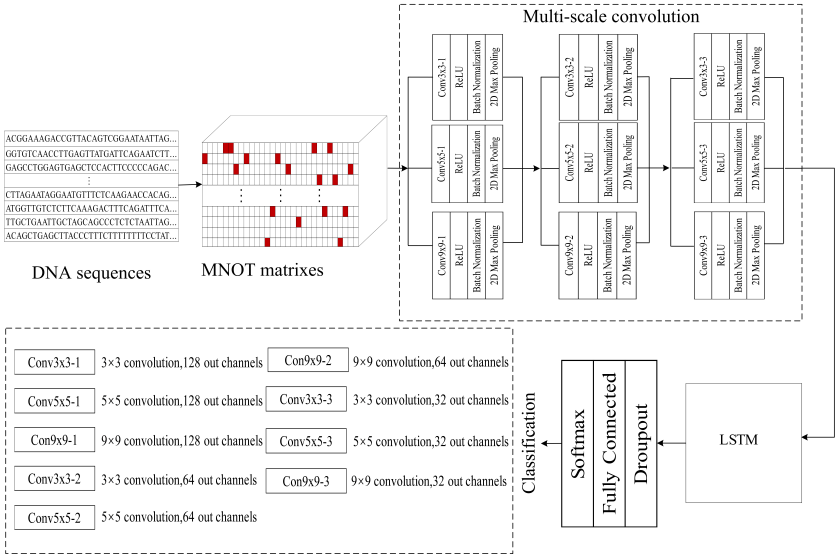


**Fig. 1.** The architecture of CNN-LSTM model.

**Multi-scale CNN.** The first part contains three multi-scale CNN layers, and each layer consists of three sublayers with different kernel sizes. Specifically, each sublayer contains convolution, rectified linear unit (ReLU) [17], batch normalization (BN) and max-pooling layer. Moreover, these sublayers separately capture different local features of the DNA sequences and produce multi-scale outputs. In light of this, we use the average of the three outputs as inputs for the subsequent layers. In addition, fused features from CNN layers with multiple kernels sizes will be more distinguishable than that from single convolutional kernel.

Here, the multi-scale CNN layers are introduced, which play an important role when processing DNA sequence features. After inputting the matrix $Z_{MNOH}$, the implementation of the convolutional part is as follows:

$$X_{i,k} = \sum_{j=1}^{m}\sum_{l=1}^{4^h} z_{i+j,l} M_{k,j,l} + b_k \tag{4}$$

Where $i \in [1, n+m-h]$ and $k \in [1, d]$, $d$ is the number of convolution kernels, $M$ is the kernel of convolution, $b_k$ represents the bias term, $m$ represents the size of the convolutional kernel, and $h$ represents the degree of MNOH.

We then feed $X$ into ReLU [17] activation function to alleviate the problem of overfitting. Next, we add BN to speed up the training of the model and improve the prediction accuracy of the model. The final layer is a max-pooling layer. This layer only picks out the maximum value of its respective previous layer outputs. The function of the max-pooling process is reducing the amount of computation.

**LSTM network layer.** As we know, the TFBS is a fragment in the DNA sequence. Hence, the TFBS prediction can be interpreted as a sequence prediction problem. In recent years, LSTM networks have been widely used for sequence tasks, such as speech recognition [18] and language translation tasks [19].

$$\begin{aligned} f_t &= \sigma(W_f[h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\ C_t &= f_t C_{t-1} + i_t tanh(W_c[h_{t-1}, x_t] + b_C) \\ o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\ h_t &= o_t tanh(C_t) \end{aligned} \tag{5}$$

The process of LSTM is above. Where $x_t$ is the current input, $i_t$ is the input gate at time $t$, $h_t$ is the hidden state, $C_t$ is the cell state at time $t$, $f_t$ is the forget gate, $\sigma$ is the sigmoid function and the $o_t$ is the output gate, $b_f$, $b_i$, $b_C$ and $b_o$ is bias terms. Moreover, the weight matrix $W$ subscripts have the obvious meaning. For instance, $W_f$ is the forget gate matrix, $W_i$ is the input gate matrix.

**Fc layer.** The final part contains the fc layer with a dropout [20] regularization and a softmax activation function. The dropout is used to randomly mask portions of its output to avoid overfitting and softmax function is used to transform the output into a probability distribution over two classes.

### 2.4 Evaluation metric

In this study, we choose Sensitivity (Sen), Specificity (Spe), Accuracy (Acc) and the Matthews correlation coefficient (MCC) to evaluate the performance of the method [21], which are defined as follows:

$$\begin{aligned} Sen &= TP/(TP+FN) \\ Spe &= TN/(TN+FP) \\ Acc &= (TP+TN)/(TP+FP+TN+FN) \\ MCC &= \frac{(TP \times TN - FP \times FN)}{\sqrt{(TP+FP) \times (TN+FN) \times (TP+FN) \times (TN+FP)}} \end{aligned} \tag{6}$$

Where $TP, TN, FN$ and $FP$ means true positives, true negatives, false negatives and false positives.

We further employ another evaluation metric, the area under the receiver operating characteristic curve (ROC), called AUC [22]. The value of AUC is between 0 and 1. The closer AUC tends to 1, the better the predictor.

## 3 Results

### 3.1 Comparisons among CNN-LSTM, CNN, and LSTM

Recently, CNN and LSTM have been adapted to the task of predicting TFBS. To demonstrate the efficacy of CNN-LSTM, we compare it with CNN and LSTM. Note that all of the three models use the simplest one-hot encoding method. Moreover, we randomly select five datasets from all 690 ChIP-seq datasets, namely A549, GM12878, Hela, Hep-G2 and H1-hESC, as benchmark datasets. For each dataset, we separately train three models on the corresponding training dataset, and evaluate the performances of these models on the test datasets. In the remaining experiments, we take the same setting which afore-mentioned.

**Table 1.** Ablation study of CNN-LSTM, CNN and LSTM.

| Dataset | Model | Sen(%) | Spe(%) | Acc(%) | MCC | AUC |
|---|---|---|---|---|---|---|
| A549 | CNN-LSTM | 59.3 | **82.0** | **76.1** | **0.518** | **0.831** |
| | CNN | 64.7 | 79.8 | 72.7 | 0.449 | 0.804 |
| | LSTM | **66.5** | 44.1 | 55.6 | 0.109 | 0.619 |
| GM12878 | CNN-LSTM | 67.2 | **77.3** | **77.2** | **0.545** | **0.854** |
| | CNN | 69.5 | 69.3 | 74.1 | 0.480 | 0.826 |
| | LSTM | **76.6** | 59.8 | 60.7 | 0.213 | 0.645 |
| Hela | CNN-LSTM | 62.5 | **84.6** | **80.3** | **0.604** | **0.801** |
| | CNN | 66.9 | 80.7 | 69.9 | 0.387 | 0.773 |
| | LSTM | **75.7** | 33.9 | 56.2 | 0.112 | 0.567 |
| Hep-G2 | CNN-LSTM | 72.6 | **80.9** | **76.8** | **0.617** | **0.844** |
| | CNN | 75.1 | 76.9 | 71.5 | 0.536 | 0.781 |
| | LSTM | **76.9** | 68.5 | 67.3 | 0.487 | 0.672 |
| H1-hESC | CNN-LSTM | **82.2** | **77.7** | **80.0** | **0.599** | **0.885** |
| | CNN | 79.2 | 71.4 | 75.5 | 0.481 | 0.793 |
| | LSTM | 70.5 | 72.8 | 68.4 | 0.364 | 0.619 |

From Table 1, we find that CNN-LSTM performs better than both CNN and LSTM. As for CNN, the four evaluation metrics (*i.e.*, Spe, Acc, MCC and AUC) of CNN-LSTM are 4.9%, 5.4%, 11.0% and 4.8% higher on average than CNN on the five datasets, respectively. Although the Sen of CNN-LSTM are lower than CNN on four out of five datasets (i.e., A549, GM12878, Hela, Hep-G2), the corresponding Spe, Acc, MCC and AUC values of CNN-LSTM are relatively higher. The underlying reason is that too many false positive example in CNN. The similar observations can also be obtained on LSTM. For example, the CNN-LSTM on Hela dataset achieves a Spe of 84.6%, an Acc of 80.3%, a MCC of 60.4% and an AUC of 80.1%, which are 50.7%, 24.1%, 49.2% and 23.4%, respectively, higher than LSTM.

To further investigate the effectiveness of the CNN-LSTM, we implement three models on all 690 ChIP-seq datasets. Fig. 2 summarizes the distribution of AUCs of these models. The red dotted line indicates the median value of AUCs.

It is easy to see that the distribution of the AUCs of CNN-LSTM on benchmark datasets is significantly higher than both CNN and LSTM. Moreover, it can be clearly observed from Fig. 2 that the median AUC of CNN-LSTM is 0.812, while the median AUC of CNN is 0.764 and LSTM is 0.527. Considering that the benchmark datasets contain 690 ChIP-seq datasets, these experimental results show the superiority of the CNN-LSTM we proposed.

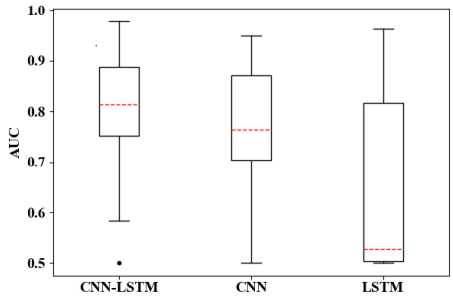


**Fig. 2.** The distribution of AUCs on 690 ChIP-seq datasets.

These findings indicate that CNN-LSTM can effectively improve the prediction accuracy for predicting TFBS in this study. One side, CNN has made great achievements in task of predicting TFBS, by which the binding preferences of transcription factor can be well characterized. Moreover, CNN layers with kernel size of 24 scans the input sequence, which aims to extract features of the TFBS. On the another side, LSTM has proved to be effective in the processing of sequence features. The LSTM network layer can take its advantages to deal with sequence features to predict the TFBS in DNA sequences.

### 3.2 MNOH helps to improve the prediction performance

In this section, we will choose the optimal degree $h$ of MNOH and prove the efficacy of MNOH. Specifically, we employ MNOH with four different degrees of MNOH, *i.e.* 1, 2, 3 and 4, denoted as 1NOH, 2NOH, 3NOH and 4NOH, respectively, to extract features from DNA sequences, which are used as the inputs of CNN-LSTM. Therefore, these CNN-LSTM models with inputs generated by different $h$ are denoted as C-L-1NOH, C-L-2NOH, C-L-3NOH, and C-L-4NOH. Similarly, we select the five datasets above as benchmark datasets. Table 2 summarizes the contrast results of the models on five datasets.

According to Table 2, we can draw two mainly conclusions as follows.

Firstly, 1NOH has the worst results. Using A549 dataset as an example, in terms of Acc, MCC and AUC three metrics, 2NOH achieves 1.7%, 2.8% and 2.1% better than 1NOH respectively. 3NOH is 1.0%, 0.8% and 1.8% better than 1NOH. 4NOH achieves 0.7%, 0.2% and 1.0% respectively, better than 1NOH. Similar observations can also be obtained on other datasets. The underlying cause of this phenomenon is that the higher $h$, the more comprehensive the sequence characteristics obtained. For example, 2NOH considers the correlation between nucleotides in two adjacent positions.

**Table 2.** Ablation study of MNOH.

| Dataset | Model | Sen (%) | Spe (%) | Acc (%) | MCC | AUC |
|---|---|---|---|---|---|---|
| A549 | C-L-1NOH | 59.3 | **82.0** | 76.2 | 0.518 | 0.831 |
| | C-L-2NOH | **62.0** | 79.6 | **77.9** | **0.546** | **0.852** |
| | C-L-3NOH | 60.3 | 81.1 | 77.2 | 0.526 | 0.849 |
| | C-L-4NOH | 61.6 | 80.1 | 76.9 | 0.520 | 0.841 |
| GM12878 | C-L-1NOH | 67.2 | 77.3 | 77.2 | 0.545 | 0.854 |
| | C-L-2NOH | 69.9 | **89.4** | **80.8** | **0.604** | **0.872** |
| | C-L-3NOH | **78.1** | 82.6 | 80.4 | 0.577 | 0.863 |
| | C-L-4NOH | 75.9 | 80.3 | 78.2 | 0.582 | 0.867 |
| Hela | C-L-1NOH | 62.5 | **84.6** | 80.3 | **0.604** | 0.801 |
| | C-L-2NOH | **74.4** | 79.2 | **83.4** | 0.506 | **0.822** |
| | C-L-3NOH | 70.8 | 77.5 | 82.2 | 0.483 | 0.819 |
| | C-L-4NOH | 67.8 | 82.0 | 83.5 | 0.503 | 0.824 |
| Hep-G2 | C-L-1NOH | **72.6** | 80.9 | 76.8 | 0.617 | 0.844 |
| | C-L-2NOH | 66.3 | **83.6** | **78.6** | **0.635** | **0.857** |
| | C-L-3NOH | 70.2 | 81.8 | 77.1 | 0.621 | 0.851 |
| | C-L-4NOH | 67.4 | 82.5 | 77.8 | 0.629 | 0.855 |
| H1-hESC | C-L-1NOH | **82.2** | 77.7 | 80.0 | 0.599 | 0.885 |
| | C-L-2NOH | 76.1 | **83.5** | **81.4** | **0.613** | **0.898** |
| | C-L-3NOH | 77.8 | 82.0 | 81.1 | 0.610 | 0.893 |
| | C-L-4NOH | 79.2 | 81.5 | 80.7 | 0.604 | 0.889 |

Secondly, 2NOH is a better choice in this study for encoding the DNA sequences. By revisiting Table 2, we observe that when $h$ is higher than 2, the performance is not significantly improved, even deteriorate. For instance, 2NOH achieves the prediction results with an Acc of 83.4%, a MCC of 50.6% and an AUC of 82.2%, which are 1.2%, 2.3% and 0.3%, respectively, better than 3NOH on the Hela dataset. We speculate that as $h$ increases, the prediction effect will deteriorate. The main reason for this phenomenon is that MNOH takes into account redundant information as $h$ increases. Take the 4NOH as an example, a sequence, with the length of $n$, is encoded into a $(n-3) \times 256$ matrix, but 99.6% of the elements in the matrix are 0. This means the matrix will be too sparse and the prediction performance will be poor. Therefore, the more the relationship between nucleotides is considered, the prediction cannot be continuously improved. Based on conclusions above, we decide to use 2NOH in our study.

### 3.3 Multi-scale convolution outperforms in prediction

In order to further explore the usefulness of multi-scale CNN layer, we compare the performances between MCNN-LSTM and CNN-LSTM on 5 datasets. Here, MCNN-LSTM denotes the CNN-LSTM, which contains multi-scale CNN layers. Note that both of the methods are based on 2NOH. Moreover, CNN-LSTM use only a fixed convolutional kernel size 24 to predict TFBS in DNA sequences, while MCNN-LSTM uses different convolutional layers with kernel size (3, 5 and 9) to capture multiple sequence features. Fig. 3 demonstrates the performance comparison between the proposed MCNN-LSTM and CNN-LSTM. Fig. 4 depicts the ROC curves along with the AUCs, listed in Fig. 3.

As shown in these figures, it is straightforward to see that MCNN-LSTM outperform the CNN-LSTM. Compared with CNN-LSTM, MCNN-LSTM achieves

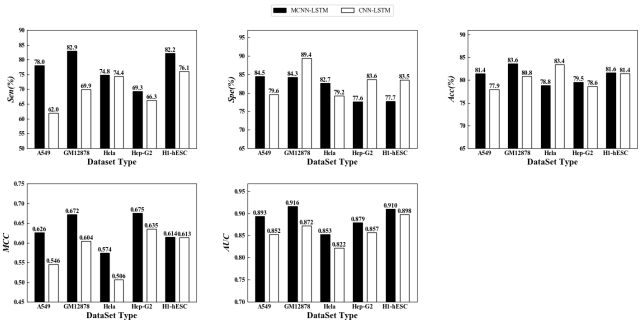


**Fig. 3.** Performance comparisons between MCNN-LSTM and CNN-LSTM.

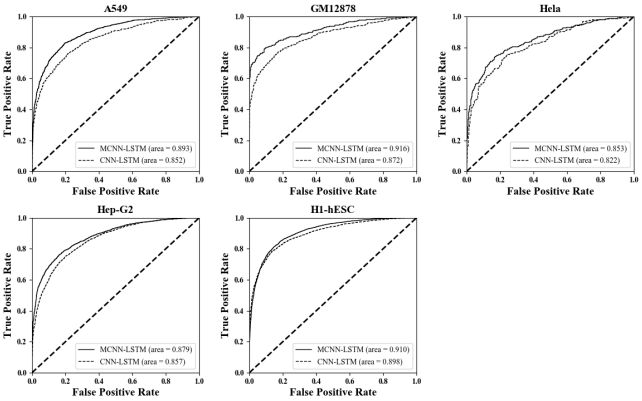


**Fig. 4.** The ROC curves of MCNN-LSTM and CNN-LSTM.

the better values of Sen, Acc, MCC and AUC. For example, the four evaluation metrics of MCNN-LSTM achieve 78.0%, 81.0%, 63.2% and 89.0% on average, which are 7.7%, 0.6%, 5.1% and 3.0% higher than those of CNN-LSTM. In addition, it cannot escape our notice that the Spe value of MCNN-LSTM are lower than that for CNN-LSTM on GM12878, Hep-G2 and H1-hESC datasets. Maybe since that CNN-LSTM tends to predict positive samples as negatives.

According to the figures above, it is obvious that the multi-scale convolutional layer is effective to predict TFBS in DNA sequences. This phenomenon can be explained as follows. The binding lengths of the transcription factor are different, hence the performance of the model with a fixed-size convolutional kernel is lower than that of the model with a multiple size convolutional kernel. Specifically, when the model scans DNA sequences with multi-scale convolution, the features of TFBS tend to be more comprehensively captured.

### 3.4 Comparisons with existing predictors

In this section, we implement a new TFBS predictor, called DeepTF, by combining the new encoding method MNOH with MCNN-LSTM. To demonstrate

the effectiveness of DeepTF, we compare it with the existing sequence-based predictors, including DeepBind [9] and CNN-Zeng [10]. Table 3 summarizes the prediction performance of DeepTF, DeepBind and CNN-Zeng on five datasets including A549, GM12878, Hela, Hep-G2 and H1-hESC dataset, selected in section 3.1. Note that we separately train the models on the train dataset of each dataset, and evaluate the performances on the corresponding test datasets.

As shown in Table 3, it is obviously that DeepTF achieves better performance than DeepBind and CNN-Zeng with respect to all of the three evaluation metrics, *i.e.*, Acc, MCC and AUC. Compared to DeepBind, DeepTF gets 1.0%, 2.0%, and 1.1% increases on A549 dataset in terms of Acc, MCC and AUC, respectively. Moreover, the Sen, Acc, MCC and AUC of DeepTF are, respectively, 2.1%, 0.1%, 0.4% and 0.3% higher than the values measured for CNN-Zeng on GM12878 datasets. As another example, the average AUCs are also 1.1% and 0.9%, respectively, higher than the corresponding methods on five datasets.

**Table 3.** Results comparisons of DeepTF, DeepBind and CNN-Zeng.

| Dataset | Model | Sen(%) | Spe(%) | Acc(%) | MCC | AUC |
|---|---|---|---|---|---|---|
| A549 | DeepTF | **78.0** | 84.5 | **81.4** | **0.626** | **0.893** |
| | DeepBind | 76.2 | 84.2 | 80.4 | 0.606 | 0.882 |
| | CNN-Zeng | 74.9 | **86.4** | 81.0 | 0.615 | 0.874 |
| GM12878 | DeepTF | **82.9** | 84.3 | **83.6** | **0.672** | **0.916** |
| | DeepBind | 73.4 | **90.0** | 82.5 | 0.642 | 0.904 |
| | CNN-Zeng | 80.8 | 86.0 | 83.5 | 0.668 | 0.913 |
| Hela | DeepTF | **74.8** | 82.7 | **78.8** | **0.574** | **0.853** |
| | DeepBind | 62.7 | **90.1** | 78.4 | 0.547 | 0.844 |
| | CNN-Zeng | 63.0 | 89.1 | 77.8 | 0.538 | 0.839 |
| Hep-G2 | DeepTF | 69.3 | **77.6** | **79.5** | **0.675** | **0.879** |
| | DeepBind | **71.6** | 73.5 | 76.2 | 0.658 | 0.866 |
| | CNN-Zeng | 66.4 | 69.7 | 77.8 | 0.671 | 0.877 |
| H1-hESC | DeepTF | **82.2** | 77.7 | **81.6** | **0.614** | **0.910** |
| | DeepBind | 79.3 | 69.4 | 81.6 | 0.591 | 0.897 |
| | CNN-Zeng | 75.5 | **78.6** | 79.5 | 0.602 | 0.901 |

To further investigate the effectiveness of our predictor, we train the three predictors on all 690 ChIP-seq datasets. Fig. 5 is the box plots of all AUCs of DeepTF and two other predictors. The red dotted line represents the median value of AUCs for all 690 datasets. Specifically, the median AUC of DeepTF is 0.880, which are 1.5% and 0.8% higher than DeepBind and CNN-Zeng, respectively, which indicates that DeepTF is superior to other predictors in terms of median AUC. Moreover, our predictor achieves the maximum value of the AUCs similar to the other two predictors, but the number of the AUCs close to 0.6 is less than both DeepBind and CNN-Zeng, which means that the DeepTF we proposed has good generalization ability and adaptability to different samples.

## 4 Conclusions

In this study, we propose a new sequence-based TFBS predictor, called DeepTF, by combining a new encoding method, called MNOH, with a new deep-learning

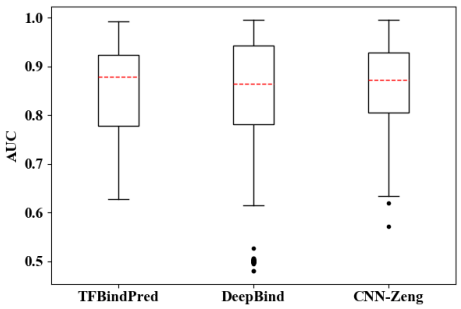


**Fig. 5.** The distribution of median AUCs on 690 ChIP-seq datasets.

model, called MCNN-LSTM. By comparison with several existing sequence-based TFBS predictors, on the 690 ChIP-seq datasets, the efficacy of the proposed predictor has been demonstrated. The superior performance of DeepTF is mainly attributed to MNOH and MCNN-LSTM. Specifically, MNOH takes account of the correlation between nucleotides in adjacent positions; MCNN-LSTM uses multi-scale convolution layers for feature processing, and then uses LSTM as a recurrent model that recognizes TFBS in DNA sequences.

Although DeepTF achieves some progress, there is still room to further improve its performance due to the following two points. First, we only use three multi-scale convolution layers in MCNN-LSTM model due to GPU memory limitations. In our future work, we try to use more multi-scale convolution layers for enhancing the performance of MCNN-LSTM. Second, DeepTF is specifically designed to predict TFBS from DNA sequences. In the future, we will further investigate the applicability of DeepTF to other types of binding site prediction problems, such as RNA-binding sites [23], ATP-binding sites [24].

**Acknowledgments**
This work was supported by the National Natural Science Foundation of China (No. 61772273, 61373062) and the Fundamental Research Funds for the Central Universities (No. 30918011104).

## References

1. Lee, D., Gorkin, D.U., Baker, M., Strober, B.J., Asoni, A.L., McCallion, A.S., Beer, M.A.: A method to predict the impact of regulatory variants from dna sequence. Nature genetics **47**(8), 955 (2015)
2. Kharchenko, P.V., Tolstorukov, M.Y., Park, P.J.: Design and analysis of chip-seq experiments for dna-binding proteins. Nature biotechnology **26**(12), 1351 (2008)
3. Ji, H., Jiang, H., Ma, W., Johnson, D.S., Myers, R.M., Wong, W.H.: An integrated software system for analyzing chip-chip and chip-seq data. Nature biotechnology **26**(11), 1293 (2008)
4. Siggers, T., Gordan, R.: Protein–dna binding: complexities and multi-protein codes. Nucleic acids research **42**(4), 2099–2111 (2013)
5. Fletez-Brant, C., Lee, D., McCallion, A.S., Beer, M.A.: kmer-svm: a web server for identifying predictive regulatory sequence features in genomic data sets. Nucleic acids research **41**(W1), W544–W556 (2013)

6. Wong, K.C., Chan, T.M., Peng, C., Li, Y., Zhang, Z.: Dna motif elucidation using belief propagation. Nucleic acids research **41**(16), e153–e153 (2013)
7. Ghandi, M., Lee, D., Mohammad-Noori, M., Beer, M.A.: Enhanced regulatory sequence prediction using gapped k-mer features. PLoS computational biology **10**(7), e1003711 (2014)
8. Nutiu, R., Friedman, R.C., Luo, S., Khrebtukova, I., Silva, D., Li, R., Zhang, L., Schroth, G.P., Burge, C.B.: Direct measurement of dna affinity landscapes on a high-throughput sequencing instrument. Nature biotechnology **29**(7), 659 (2011)
9. Alipanahi, B., Delong, A., Weirauch, M.T., Frey, B.J.: Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. Nature biotechnology **33**(8), 831 (2015)
10. Zeng, H., Edwards, M.D., Liu, G., Gifford, D.K.: Convolutional neural network architectures for predicting dna–protein binding. Bioinformatics **32**(12), i121–i127 (2016)
11. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436 (2015)
12. Hassanzadeh, H.R., Wang, M.D.: Deeperbind: Enhancing prediction of sequence specificities of dna binding proteins. In: 2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM). pp. 178–183. IEEE (2016)
13. Zhou, J., Troyanskaya, O.G.: Predicting effects of noncoding variants with deep learning–based sequence model. Nature methods **12**(10), 931 (2015)
14. Siebert, M., Söding, J.: Bayesian markov models consistently outperform pwms at predicting motifs in nucleotide sequences. Nucleic acids research **44**(13), 6055–6069 (2016)
15. Salekin, S., Zhang, J.M., Huang, Y.: Base-pair resolution detection of transcription factor binding site by deep deconvolutional network. Bioinformatics **34**(20), 3446–3453 (2018)
16. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
17. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks proceedings of the fourteenth international conference on artificial intelligencs & statistics. In: AISTATS. vol. 130, p. 297 (2011)
18. Graves, A., Jaitly, N., Mohamed, A.r.: Hybrid speech recognition with deep bidirectional lstm. In: 2013 IEEE workshop on automatic speech recognition and understanding. pp. 273–278. IEEE (2013)
19. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in neural information processing systems. pp. 3104–3112 (2014)
20. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research **15**(1), 1929–1958 (2014)
21. Hu, J., Zhou, X., Zhu, Y.H., Yu, D.J., Zhang, G.: Targetdbp: Accurate dna-binding protein prediction via sequence-based multi-view feature learning. IEEE/ACM transactions on computational biology and bioinformatics (2019)
22. Zhu, Y.H., Hu, J., Song, X.N., Yu, D.J.: Dnapred: accurate identification of dna-binding sites from protein sequence by ensembled hyperplane-distance-based support vector machines. Journal of chemical information and modeling (2019)
23. Ren, H., Shen, Y.: Rna-binding residues prediction using structural features. BMC bioinformatics **16**(1), 249 (2015)
24. Chen, K., Mizianty, M.J., Kurgan, L.: Atpsite: sequence-based prediction of atp-binding residues. In: Proteome science. vol. 9, p. S4. BioMed Central (2011)