IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 23, 2018, accepted June 13, 2018, date of publication June 18, 2018, date of current version July 6, 2018.

*Digital Object Identifier 10.1109/ACCESS.2018.2848847*

# DeepSS: Exploring Splice Site Motif Through Convolutional Neural Network Directly From DNA Sequence

**XIUQUAN DU[1,2], YU YAO[1], YANYU DIAO[1], HUAIXU ZHU[1], YANPING ZHANG[1,2], AND SHUO LI[3,4]**

[1]Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, Hefei 230601, China
[2]School of Computer Science and Technology, Anhui University, Hefei 230601, China
[3]Department of Medical Imaging, Western University, London, ON N6A 3K7, Canada
[4]Digital Imaging Group, Western University, London, ON N6A 3K7, Canada

Corresponding author: Xiuquan Du (dxqllp@163.com)

This work was supported in part by the National Science Foundation of China under Grant 61673020, in part by the Provincial Natural Science Research Program of Higher Education Institutions of Anhui province under Grant KJ2016A016, and in part by the Anhui Provincial Natural Science Foundation under Grant1708085QF143.

**ABSTRACT** Splice sites prediction and interpretation are crucial to the understanding of complicated mechanisms underlying gene transcriptional regulation. Although existing computational approaches can classify true/false splice sites, the performance mostly relies on a set of sequence- or structure-based features and model interpretability is relatively weak. In viewing of these challenges, we report a deep learning-based framework (DeepSS), which consists of DeepSS-C module to classify splice sites and DeepSS-M module to detect splice sites sequence pattern. Unlike previous feature construction and model training process, DeepSS-C module accomplishes feature learning during the whole model training. Compared with state-of-the-art algorithms, experimental results show that the DeepSS-C module yields more accurate performance on six publicly donor/acceptor splice sites data sets. In addition, the parameters of the trained DeepSS-M module are used for model interpretation and downstream analysis, including: 1) genome factors detection (the truly relevant motifs that induce the related biological process happen) via filters from deep learning perspective; 2) analyzing the ability of CNN filters on motifs detection; 3) co-analysis of filters and motifs on DNA sequence pattern. DeepSS is freely available at http://ailab.ahu.edu.cn:8087/DeepSS/index.html.

**INDEX TERMS** Splice sites, convolutional neural network, feature extraction, motifs.

## I. INTRODUCTION

Splice sites are the boundaries between introns and exons in DNA sequence. The more accurately a splice site can be located, the easier it locates the gene in a DNA sequence [1]. However, dinucleotide GT/AG often align to several error places of a reference genome and the orientations of splice sites are not fixed, it is an extremely challenging task for biologists to identify splice sites. However, experimental methods are costly and time-consuming for splice sites identification. Hence, many computational methods have been proposed, which mostly use a three-stage process to first construct a set of candidate features, then select a most effective feature subset and finally feed them into a machine learning model for final classification.

Feature extraction is the first but important step for classification problems [2]. Many effective feature extraction methods have been proposed for feature construction with the original information as much as possible from DNA sequence, such as MM1 (1-order Markov model) encoding [3], MCM (Markov Chain Model) encoding [4], [5], DM (Distance Measure) encoding [6], FDTF encoding [7], Baye's feature mapping [8], nucleotide density-based encoding [9], HSplice encoding [10] and so on. The existing feature extraction approaches mostly employ features combination based on different statistical strategies to describe splice sites characteristic. For example, Meher *et al.* [10] and Meher *et al.* [11] combined features based on position, dependency, composition and di-nucleotide association difference score features. Pashaei *et al.* [12] introduced a modified

nucleotide encoding method which calculated MM1 [3] and DM [6] for each DNA sequence and then applied AdaBoost classifier for prediction. Golam Bari *et al.* [9] utilized density information of each nucleotide along with positional information and chemical property for splice sites prediction.

Given a set of candidate features, different feature representations can entangle and hide more or less the explanatory dependency factors behind the data [13]. Thus, a non-redundant feature subset that express original data sufficiently is a preprocessing step for every classification task [14]. It can reduce feature dimension, memory allocation and computational time [15]. For example in [11], F-score [16] was used to select the important features among 344 features.

After feature extraction, different machine learning approaches, such as Support Vector Machine (SVM) [3], [10], [17], [18], Random Forest (RF) [11], Decision Trees (DT) [19], Native Bayesian (NB) [20], Markov model [21] and AdaBoost [22] have been used to discriminate true and false splice sites. Among them, SVM [3], [10], [17] models and related kernel methods are most frequently used due to their high accuracy and capability of high-dimensional large data sets. For example, Baten *et al.* [3] employed MM1 to extract the features from splice sites sequences and sent them into SVM to distinguish true splice sites and false splice sites. Zhang *et al.* [17] utilized linear SVM algorithm with a Bayes kernel (SVM-B) to discriminate consensus dinucleotide GT/AG from pseudo splice sites. In another study, orthogonal encoding, codon usage and sequential information were also successfully used for splice sites prediction through SVM [18]. However, although SVM classifier is frequently used for splice sites prediction and achieves high performance, some parameters of SVM classifier such as penalty parameters, kernel type, and kernel parameters, must be tuned. Parameter tuning is time-consuming. As an alternative approach, RF algorithm which consists of ensemble of several tree classifiers is also widely used for this problem. Meher *et al.* [11] presented a sequence encoding approach based on the adjacent di-nucleotide dependencies and demonstrated that RF achieved higher accuracy than SVM, ANN (Artificial Neural Network), Bagging, Boosting, Logistic Regression, KNN and NB classifiers. Decision trees had also been proposed by Lopes *et al.* [19] to build discriminative models between real and pseudo splice sites. NB along with an automated feature generation program had also been developed by Kamath *et al.* [20] for the splice sites prediction. To better discrimination splice sites, many Markov models (MM) have been proposed. Zhang *et al.* [21] designed a length-variable Markov Model (LVMM) which could achieve higher accuracy and keep the low time cost. Unfortunately, it is difficult to determine the method's threshold parameters. In addition, Pashaei *et al.* [22] proposed a hybrid algorithm by combining AdaBoost with FDDM encoding method. This method produced an improvement performance compared with SVM, LVMM and DM2-AdaBoost.

The effectiveness of these algorithms largely depends on feature construction and the key issue is that how to define a space of potentially effective features. However, features extraction step is often done by domain experts and how the selected features to determine prediction performance remain unknown and undefined. Shortly speaking, feature extraction by manual operation often leads domain-specific feature representation either incomplete or hard to translate into effective features. Furthermore, simply concatenating different numerical type features together may result in one high-dimensional feature space and difficulties for the following machine learning process. As this case in point, machine learning algorithms that employ SVM as splice sites predictor mostly add a feature selection step to avoid dimension disaster. Unfortunately, sometimes even feature selection methods do not solve the vast number of possible feature combinations [23]. Meanwhile, frequent noise brought by feature construction from the observed low-level data may make the subsequent classifiers to learn wrong knowledge. What is worse, separation of feature extraction and model training limits classifier's capacity in capturing and representing higher-order nonlinear relationships. Lastly, although machine learning approaches can achieve state-of-the-art performances in splice sites prediction task, splice site sequence pattern discovery from a black-box chartered machine learning model is largely unknown. A deeper understanding of the mechanisms underlying the splice sites-associated function and evolution has been strongly urged.

In general, the existing computational methods still encounter numerous issues, such as their inability to extract and organize the discriminative information from raw data, over-fitting, inevitable separation of features extraction and model training, difficulty definition of a space of effective features and challengeable discovery of splice site sequence pattern. As a consequence, there is an urgent demand for a more powerful predictor that can construct effective features automatically, achieve high classification performance and detect splice site sequence pattern behind DNA sequences simultaneously.

Recent technological advances in machine learning have enabled deep learning for feature representation. Briefly, deep learning utilizes hierarchical architectures to represent global high-level abstract features from the raw data, which encapsulates highly complicated functions in the process. It is an emerging approach and has achieved remarkable results in image processing [24], transfer learning [25], natural language understanding [26], speech recognition [27], [28] and most recently, it has rapidly become a methodology for resolving the sequence-based bioinformatics problems [29]–[31]. There has developed many deep learning architectures for specific applications, such as Convolutional Neural Networks (CNN) for bioinformatics problems [30], Recurrent Neural Networks (RNN) [32] for sequential data, Restricted Boltzmann Machines (RBM) [33], [34], AutoEncoders [35] and Deep Belief Networks (DBN) [36] for unsupervised learning. At present, CNN is the most successful

algorithm for biology sequences analysis among those widely used architectures.

A CNN model trains complex network with multiple layers to capture their internal structure by convolutional operations and weight sharing mechanism. This architecture can greatly reduce the number of model parameters compared with a fully connected network and the potential mismatch effects between feature extraction and learning classification models. Moreover, CNN allows directly training on the DNA sequence without feature extraction.

Attesting to its utility, CNN have been successfully applied to predict specificities of DNA- and RNA-binding proteins [29] or epigenetic marks and the effect of DNA sequence alterations [30], [31]. One representative application of CNN is DeepBind [29] and it can generate new DNA and RNA binding sites prediction, discovery novel sequence motifs and identify functional SNVs from diverse experimental data sets. With their initial successes, convolutional architectures have been extended and applied to a range of tasks in regulatory genomics. For example, Zhou and Troyanskaya [31] constructed a deep learning-based algorithm framework which is widely called DeepSEA to predict chromatin marks from DNA sequence. In a similar vein, Min *et al.* [37] utilized convolution neural network to predict enhancers, and experimental results demonstrated that the built model DeepEnhancer had superior efficiency and effectiveness than the gapped k-mer support vector machine (gkm-SVM). Umarov and Solovyev [38] built a CNN model to analyze sequence characteristics of prokaryotic and eukaryotic promoters and obtained an excellent performance between promoters and non-promoter sequences. All these methods suggest that deep learning is a powerful tool in genomics studies, stimulating us to ask the question whether splice sites can be identified and splice sites sequence pattern can be detected merely utilizing the sequence information by adopting a deep learning framework.

In this study, a CNN model DeepSS, which could capture splice sites internal pattern by applying convolutional operations and weight sharing mechanism, has been proposed. Compared with the other existing approaches, DeepSS consists of DeepSS-C module to classify splice sites and DeepSS-M module to detect splice sites sequence pattern. DeepSS has the following merits:

1) It is a deep learning framework and consists of neural networks stacked together [39], [40], where the outputs of each layer are the inputs of successive layer. Such layer-by-layer learning helps to reduce the noise effects in the original input.
2) The CNN model employs stacked convolutional-pooling operations to identify predictive motifs from splice sites sequence context and two fully connected layers to model motif interactions, then to resolve splice sites prediction and motif discovery problems.
3) Unlike previous methods [10], [11] needing to define a space of potentially effective features as well as separate feature extraction and model training, DeepSS-C can capture the inherent and complex high-level features from low-layer string format DNA sequence directly, learn global representation from data at each layer and accomplish feature learning during the whole model training.
4) Previous methods did not introduce the motifs when giving a classification prediction. Our trained DeepSS-M module can detect splice sites-associated motifs by a set of learnt filters from the first convolutional layer. Owing to the extracted motifs, the principles behind splice sites can be interpreted.

## II. MATERIAL AND METHODS

In this section, three datasets in the experiments, related representation methods, classical CNN concept, DeepSS implementation details, as well as evaluation metrics are introduced.

### A. DATADESTS

A distinction is made between acceptor and donor splices sites, hence splice site datasets are split into donor set and acceptor set, and classification performance is compared separately on each subset [20]. To demonstrate our method's generality, high-quality balanced/imbalanced splice sites datasets (donor and acceptor), such as Homo sapiens (HS) and Caenorhabditis elegans (CE) are adopted. Besides, the benchmark splice sites dataset NN269 is also used to compare the performance of the proposed approach with the other splice sites prediction approaches.

#### 1) HS$^3$D

The HS$^3$D dataset [41] is composed of exons, introns and splice regions of human and contains 2796 confirmed true donor sites, 90953 confirmed false donor sites, 2880 confirmed true acceptor sites and 90353 confirmed false acceptor sites. Both donor and acceptor sequences are 140 nucleotides whereas conserved nucleotides GT at 71st and 72nd positions and conserved nucleotides AG at 69th and 70th positions, respectively. This dataset is freely available at http://www.sci.unisannio.it/docenti/rampone/.

#### 2) CE

The performance is also reported on the another dataset extracted from C_Elegans (CE) genome [42]. In case of CE, true acceptor set contains 1000 sequences and false acceptor set contains 19,000 sequences. After handling the missing label corresponding to each donor splice site sequence manually, CE donor dataset including 750 positive samples and 19250 negative samples is established. Each of acceptor/donor sequence consists of 141 nucleotides, which has conserved nucleotides AG at 60th and 61st positions and GT at 63rd and 64th positions, respectively. The true and false CE splice sites are collected from http://www.fml.mpg.de/raetsch/projects/splice.

#### 3) NN269

The NN269 dataset [43] is extracted from 269 human genes and composed of 1324 confirmed true acceptor sites, 5552 confirmed false acceptor sites, 1324 confirmed true donor sites and 4922 confirmed false donor sites. The length of acceptor sequences is 90 nucleotides whereas donor splice sites sequence has the length of 15 nucleotides. The consensus dinucleotide AG in acceptor splice sites is at positions 69 and 70 and the consensus nucleotides GT in donor splice sites is at positions 7 and 8. The acceptor and donor splice sites sequences are split into training and testing dataset. The training dataset has 1116 true acceptor, 1116 true donor, 4672 false acceptor, and 4140 false donor sequences. The testing dataset has 208 true acceptor, 208 true donor, 881 false acceptor, and 782 false donor sequences. This dataset is also available at http://www.cs.gmu.edu/~ashehu/sites/default/files/tools/EFFECT_2013/data.html.

### B. CONVOLUTIONAL NEURAL NETWORK

CNN is a type of feed-forward ANN in which the individual neurons are arranged in agree with local connectivity and parameter sharing mechanism. Without predefine any feature sets, CNN can extract features from raw inputs while keeping the number of model parameters tractable by applying a series of convolutional and pooling operations. In standard applications, a CNN has multiple pairs of convolutional-pooling layers, which are followed by one or more fully connected layers after the last pooling layer.

Convolutional operation is the heart of convolutional networks. If the weight of each neuron connected data window is fixed, all neurons within a filter only focus on one characteristic in the previous layer, whatever at different locations. More specifically, during each convolutional operation, the input is convolved with a set of K filters $W = \{w_1, \cdots w_k\}$ and subsequently biases $B = \{b_1, \ldots, b_k\}$ are added, each filter generates a new feature map $X_k^l$. These features are subjected to a non-linear transform $\sigma(\cdot)$ and the same process is repeated for every convolutional layer:

$$X_k^l = \sigma\left(W_k^{l-1} \otimes X_{k1}^{l1}\right) + b_k^{l-1} \tag{1}$$

where $W_k^{l-1}$ is the weight of convolutional filter at previous layer $l-1$, $X_{k-1}^{l-1}$ is the sub-matrix with equal size at every position in the input matrix, $b_k^{l-1}$ is bias of previous layer $l-1$, $X_k^l$ is the value after convolutional operation.

An activation function often lies after a convolutional layer and its main target is to guarantee the nonlinearity of the whole model by filtering out some unnecessary information. The most popular activation function is the rectified linear unit (ReLU) function. It filters out noise by thresholding negative values to 0 and only keeps significant signals with positive values:

$$\mathrm{ReLU}(x, b) = \begin{cases} x & if\ (x > b) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $b$ is the activation threshold, x is the activation value.

The pooling layer summarizes adjacent neurons. For example, the maximum or average operation on activity is used to result in a smoother representation of feature activities [23]. The max-pooling function is usually used to avoid over-fitting and help to abstract the features learned in the previous layers. A max-pooling layer is the common pooling technique and it takes the maximum value of the signal on non-overlapping windows for further representation.

$$Z_k = \max\left(Y_{1,k}, \cdots, Y_{n,k}\right) \tag{3}$$

where $n$ is the max-pooling window size, $k$ is motif detector.

After several convolution and pooling operations, there may be one or more fully connected layers or dense layers. The weights in these layers are no longer shared. A softmax function often acts as a nonlinear classifier which attached to the last layer. The equation is as below:

$$f_i(z) = \exp(z_i) \Big/ \sum_j \exp(z_j) \tag{4}$$

where $f_i(z)$ denotes the predicted score for class $i$.

A dropout layer is used between fully connected layers to randomly mask portions of its output to avoid over-fitting [44]. The object function for a classification network is often a cross entropy loss function:

$$H(p, q) = -\sum p \log q \tag{5}$$

where $p$ denotes a true distribution and $q$ denotes the estimated class probability.

### C. THE FRAMEWORK OF DEEPSS METHOD

In this work, we propose a CNN model DeepSS to predict splice sites only based on DNA sequences and explore important motifs from DNA sequences around splice sites. The graphical illustration of DeepSS is showed in Fig. 1.

The DeepSS model is composed of DeepSS-C and DeepSS-M modules and the two parts have the same CNN architecture. Two stacked convolution-pooling layers are designed for discovering useful motifs and learning the patterns in the data, after that, two fully connected layers are followed. The first fully connected layer is used to learn deep feature interactions and the last layer corresponds to the prediction results. The shared model structure and parameters are interpreted in detail as below:

To prepare the input for CNN architecture, raw $N$ nt long DNA sequence centered on a target splice site is encoded into a $N \times 4$ binary matrix with columns corresponding to A, G, C and T ($N$ denotes the number of nucleic acids) by one-hot encoding method. Additionally, we construct a binary mask vector of the same length. 1 represents the value unit and 0 represents the zero units for each splice site sequence.

The initial layer of our network is a convolutional layer (denoted by CONV), which consists of a set of learnable filters with equal size. Each filter scans on the encoded matrix in a manner analogous to a sliding window. It matches against

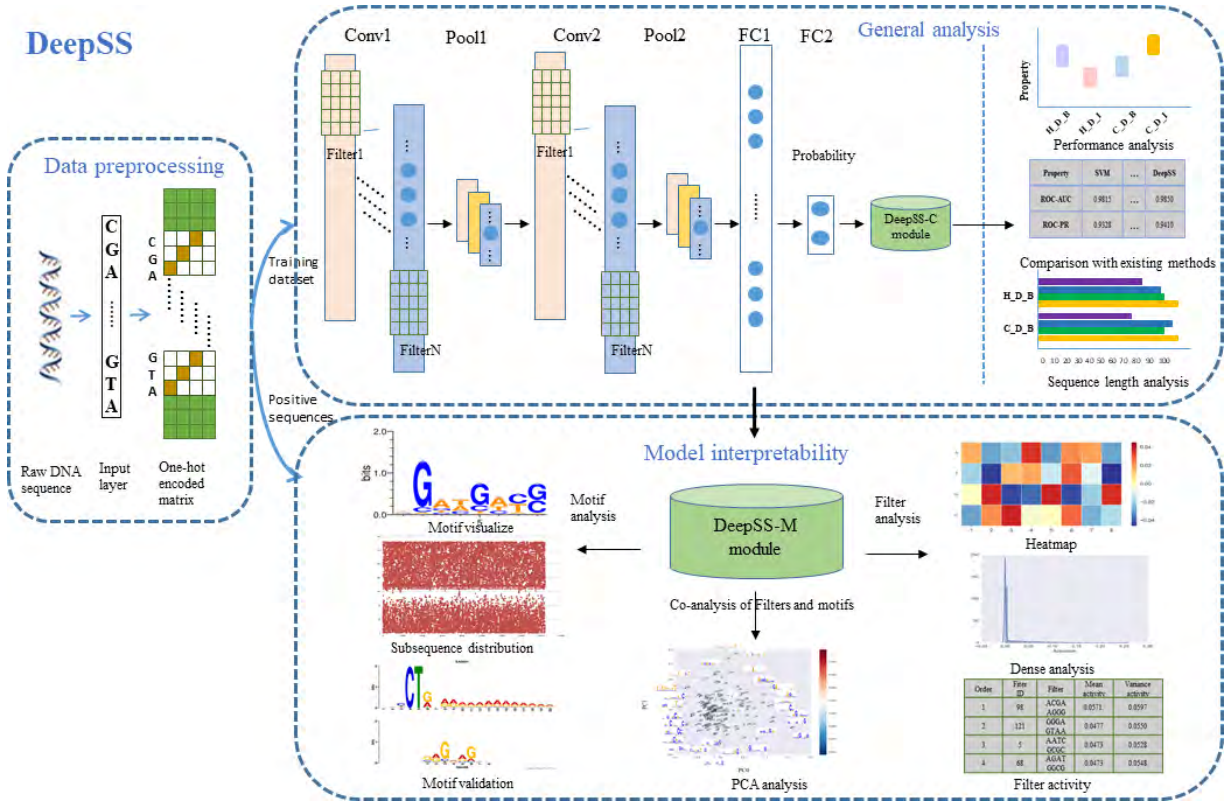



**FIGURE 1.** **Graphical illustrations of DeepSS model. DeepSS consists of two separate modules: DeepSS-C and DeepSS-M. DeepSS-C is used for discriminating true and false splice sites and DeepSS-M is trained for discovering splice site sequence pattern and some relative downstream analysis.**

each sub-matrix at every position to detect distinct motifs around target splice site. Motifs are subsequences composed of different numbers of nucleotide and are important for the recognition between splice sites and non-splice sites. The default rectified linear unit (ReLU) activation function $z = \max(0, x)$ is adopted in the convolution layer. The formula means that if activation value x is greater than 0, the related motif at fixed position has passed to the next stage, otherwise the motif in the sequence is deemed irrelevant and the relative score is zero. This layer is used to filter the unimportant features and keep only the features with scores larger than a specified threshold. The positive values correspond to the ability of the motif affect the splice sites. The higher a positive value is, the more probability it can promote dinucleotide GT/AG to be a true splice site.

The output of the convolution layer with input matrix is another matrix of size $(N - F + 1) \times K$, where $N$ denotes the length of the padded sequence, $F$ denotes the length of the filter and $K$ denotes the total number of filters. The element in the convolved matrix is essentially the value of the filter aligned to every position of the padded sequence. Considering that the length of sequence is variable, K is set as the number of the length of input sequence plus one and F is 8.

After that, a max-pooling layer (denoted by POOL) performs a downsampling operation to identify the most relevant effective features and reduce the dimension of hidden feature. In our study, for each filter, the max operator is applied within a window size of 3 and takes the maximum value in each window. Next layer is a dropout layer and the dropout probability is 0.5.

Another convolutional operation with the same number kernels of shape 1*8 is designed for high-level feature extraction. After the following max-pooling layer with pooling size 1*3 and a dropout layer with dropout probability 0.5, the outputs at all positions are concatenated and fed into the subsequent fully-connected layer (denoted by FC). Finally, the softmax layer generates the classification probability results.

In our experiments, the max number of epochs equal to 30 and the batch size is set as 50.

For our architecture, in order to determine the deep of network, we try different CNN models from 1-depth to 4- depth. Finally, 2-depth CNN model which consists of two CONV+POOL and two FC obtains the best performance. Due to the different purpose of performance evaluation and downstream analysis, the same CNN architecture is trained on different manner. Briefly, DeepSS-C with 10-fold cross validation is designed to evaluate the model's discrimination ability between splice sites and non-splice sites and also account for the correlations between sequence length and

DeepSS-C performance. In every epoch, DeepSS-C module is feed on the training set, hyper-parameters are optimized on the validation set, the final model performance and interpretations are exclusively reported on the test set. DeepSS-M module fitted on whole DNA sequences is used to explore sequence pattern that associated with splice sites. Our parameterized convolutional neural network are built using Keras [45]. It uses Tensorflow library as a backend and utilizes GPU for fast neural network training.

### D. ONE-HOT ENCODING TECHNIQUE

A DNA sequence is a string while the neurons in the network can only handle the numerical data. Hence, one-hot encoding technique [29] is adopted. It converts the string format DNA sequence to a ''one-of-four'' numerical representation. Each column of the matrix corresponds to a particular type (A, T, C, or G), and each row corresponds to the specific location of a nucleotide in the sequence. Thus, a nucleotide of a particular type (A, T, C, or G) is encoded by a binary vector with the length of 4, in which the corresponding position is 1 while the others are 0. Specifically, given a DNA sequence with $n$ nucleotides

$$S = (s_1, \cdots, s_n), \quad s_i \in \{A, T, C, G\}$$

and a motif detector of size $m$, the DNA sequence $S$ should be padded by concatenating $(m-1)$ unusual characters on either sides and then stored as an $(n+2m-2) \times 4$ array ($R$) in the following way [29]:

$$R_{i,j} = \begin{cases} 0.25 & if\ s_{i-m+1} = N\ or\ i < m\ or\ i > n-m \\ 1 & if\ s_{i-m+1} = j^{th}\ base\ in\ (A, C, G, T) \\ 0 & otherwise \end{cases} \quad (6)$$

where $m$ is the size of the motif detector and $n$ is the length of the DNA sequence, $i$ is the index of nucleotides, $j$ is the index corresponding to A, C, G, T.

### E. CROSS VALIDATION

Cross-validation procedure has been widely used to validate the effectiveness of classifiers [48]. Thus, a 10-fold cross validation procedure is applied to evaluate the DeepSS-C splice sites prediction ability and compare their performance with the other available methods. For 10-fold cross validation, the whole dataset is divided into ten subsets with approximately equal size. Every subset does not share any the same sequence and keeps the same positive/negative ratio in whole dataset. In each fold, 9 out of 10 subsets are used for training and the remaining one is used for testing. The results are averaged over 10 different train-and-test experiments.

### F. PERFORMANCE EVALUATION

In order to evaluate the performance of DeepSS-C for splice sites prediction, the average AUC_ROC is used [3]. AUC_ROC compares the classifiers' performance across the entire range of class distributions and error costs. However, because the number of splice sites and non-splicing sites in the datasets are imbalanced, this criterion is no longer adequate since the minority label would have less impact on AUC_ROC than the majority label, which could result in biased performance evaluation. To measure the imbalanced applications, the average AUC-PR score is incorporated. For imbalance problem, AUC-PR provides an appropriate evaluation. Additionally, Accuracy (ACC), Specificity (SP), Sensitivity (SN), Precision (PRE) and Matthew's correlation coefficient (MCC) are also calculated to evaluate the prediction performance of the proposed method. SN/SP is defined as the proportion of true positives/negatives that are correctly identified by the classifier. Since neither SN nor SP constitutes good measures of global accuracy, ACC and PRE are taken into account as measures. ACC is the proportion of the candidate sites that are classified correctly, which tells how well DeepSS-C can assign true sites and false sites into the right categories. ACC, SN and SP are sensitive to the class distribution of the dataset, because there are many more false splice sites than true ones. MCC gives a comprehensive assessment of the performance by incorporating both sensitivity and specificity measures. All these formulas are defined as follow:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (7)$$

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

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

$$PRE = \frac{TP}{TP + FP} \quad (10)$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (11)$$

The $TP$, $TN$, $FP$ and $FN$ show the number of true positive splice sites, true negative splice sites, false positive splice sites and false negative splice sites, respectively.

**TABLE 1. Brief description of four varied hidden layers and Depth of network assessment on HS$^3$D donor imbalanced dataset.**

| Architecture of varied depth | Time cost (s/round) |
|---|---|
| 1* (CONV-POOL)-2FC layers | 2.29 |
| 2* (CONV-POOL)-2FC layers | 2.92 |
| 3* (CONV-POOL)-2FC layers | 3.38 |
| 4* (CONV-POOL)-2FC layers | 4.84 |

## III. RESULTS

This section is organized as follows: we present depth impact on network, discriminative ability of DeepSS-C, performance comparison of DeepSS-C with the other existing methods, effect of sequence length for the performance of DeepSS-C and downstream analysis by DeepSS-M.

**TABLE 2.** The average performance of DeepSS-C with balanced acceptor/donor datasets.

| Measure | $HS^3D$ | | CE | | NN269 | |
|---|---|---|---|---|---|---|
| | Acceptor | Donor | Acceptor | Donor | Acceptor | Donor |
| *AUC-ROC* | 98.50±0.42 | 98.89±0.41 | 99.32±0.49 | 99.19±0.54 | 98.98±0.51 | 98.47±0.65 |
| *AUC-PR* | 98.22±0.70 | 98.79±0.52 | 99.27±0.63 | 99.15±0.72 | 98.87±0.66 | 98.20±0.96 |

**TABLE 3.** The average performance of DeepSS-C with imbalanced acceptor/donor datasets.

| Measure | $HS^3D$ | | CE | | NN269 | |
|---|---|---|---|---|---|---|
| | Acceptor | Donor | Acceptor | Donor | Acceptor | Donor |
| *AUC-ROC* | 98.79±0.25 | 99.02±0.32 | 99.56±0.33 | 99.47±0.32 | 99.34±0.23 | 98.43±0.44 |
| *AUC-PR* | 94.28±1.12 | 95.93±1.07 | 98.18±1.00 | 97.88±1.01 | 97.32±0.93 | 93.97±1.79 |

### A. DEPTH OF NETWORK AND TIME-CONSUMING

We vary the CNN architectures to investigate how different architectures will affect the network classification performance. The basic architecture is illustrated in Table 1. n* (CONV-POOL)-2FC means the architecture has *n* convolutional-pooling layers and 2 fully-connected layers.

To explore the impact of the network depth, we append additional convolutional-pooling layers to make the CNN deeper based on stacked (CONV-POOL) architecture. In this study, we compare totally 4 varied CNN architectures to get the best depth.

From the Fig. 2, we can see that 2* (CONV-POOL)-2FC architecture better than the other architectures (9 of the 12 datasets on AUC_ROC and AUC_PR). Note that the length of NN269 donor sequence is only 15 nucleotides, which is not long enough for the second pooling operation. In the practice model training, we remove the second pooling layer to guarantee the running of code. If the sequence length is long enough for the convolutional-pooling operation, we believe that 2*(CONV-POOL)-2FC architecture will get better performance in this dataset. When we set *n* as 3, the overall performance is slightly lower than 2* (CONV-POOL) -2FC architecture. When *n* increases to 4, the performance on seven datasets drop sharply, and the variance reached to 24%. Considering that the use of deep networks constructed with additional hidden layers was driven not by a desire to increase the model's complexity, but rather to allow for the stacked (CONV-POOL)-layer design to learn the patterns in the data, we set *n* as 2 in our final model. The mean ROC-AUC and ROC-PR improvement produced by adding 1*(CONV-POOL)-layer is statistically significant. In any case, this preliminary experiment demonstrates that our use of the deeper network does lead to a visible increase of the accuracy of predictions, and the more complex architecture is not detrimental to the performance of the splice site prediction tool, in condition that the length of input sequence is long enough for the stacked pooling operation.

We further analyze the time-performance tradeoff for different architectures. Table 1 gives the time of the different depth architecture on $HS^3D$ imbalanced dataset. We can see that deeper architecture requires more additional training time.

### B. SPLICE SITES DISCRIMINATIVE ABILITY OF DEEPSS-C

To demonstrate the ability of DeepSS-C for splice sites prediction, we evaluate DeepSS-C on three different datasets

($HS^3D$, NN269 and CE). From the collected data, it is found that there have great imbalancedness between the presence of true and false splice sites in a gene. Hence, we construct balanced and imbalanced acceptor/donor datasets for three datasets. For balanced case, the number of true and false splice sites in all datasets is kept in the ratio of 1:1 and the false splice sites are randomly selected from the available false splice sites. For imbalanced case, the ratio of positive to negative sequences is kept as 1:5. Duo to NN269 donor/acceptor datasets are divided to imbalanced training and testing datasets from original literature [43], we gather true and false samples from training and testing datasets to construct balanced NN269 acceptor/donor datasets. To assess the robustness of DeepSS-C and avoid the chance results in both balanced and imbalanced situation, we randomly generate 10 negative data. All experiments are performed 10 times with 10-fold cross validation and the average performance is reported in terms of the evaluation criteria described above. The average AUC-ROC and AUC-PR of all donor/acceptor datasets are shown in Table 2 (balanced case) and Table 3 (imbalanced case). For each experiment, DeepSS-C is trained with the same initializations. After we

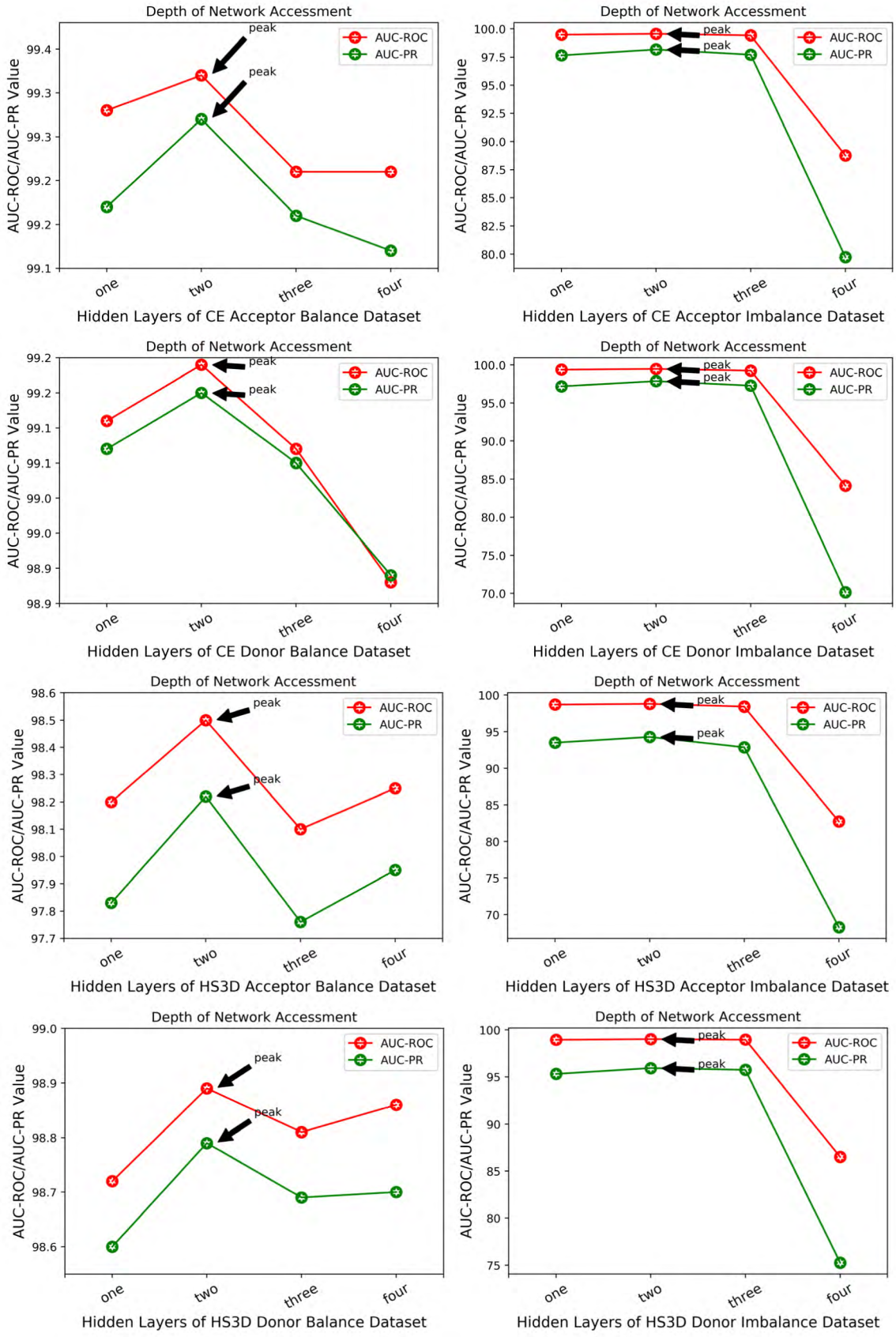



**FIGURE 2.** The performance of different depth architectures with 10-flod cross validation on 12 datasets.

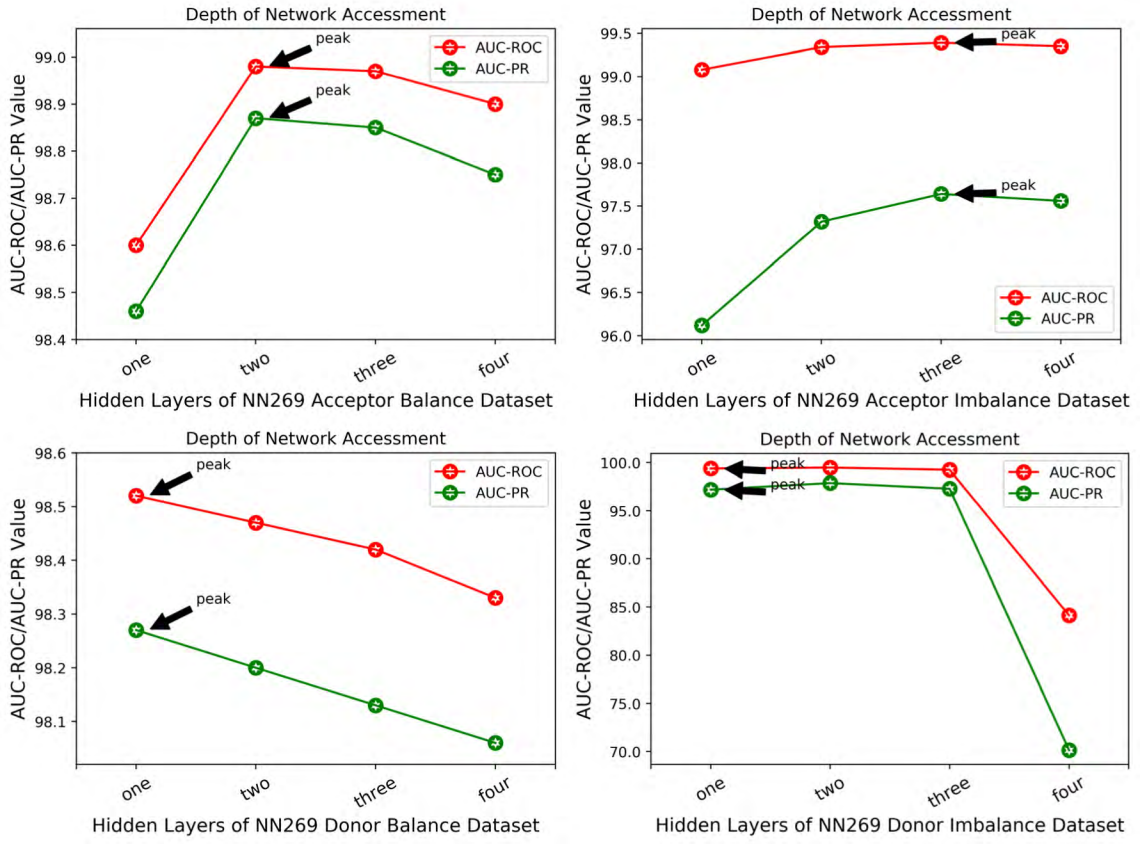



**FIGURE 2.** ***Continued.*** **The performance of different depth architectures with 10-flod cross validation on 12 datasets.**

obtain the trained model, the testing set is used to evaluate the ability of DeepSS-C for splice sites prediction.

From Table 2, it is observed that the values of AUC-ROC are between 98.47% and 99.32%; AUC-PR is between 98.20% and 99.27% in balanced acceptor/donor datasets. In particular, the AUC-ROC/AUC-PR on CE acceptor and donor splice sites dataset even reach up to 99.32%±0.49%/99.27%±0.63% and 99.19%±0.54%/99.15%±0.72%, which shows that our model gives an excellent discrimination of true and false splice sites on acceptor/donor datasets. For imbalanced situation, the AUC-ROC values are found to be higher than 98.43%, whereas the values of AUC-PR are found between 93.97% and 98.18% (Table 3). According to the data provided in Table 2 and 3, we can draw the following conclusions:

*1)* Considering both balanced and imbalanced cases, the values of AUC-PR are observed to be lower as compared with the corresponding AUC-ROC.

*2)* Furthermore, it is observed that the differences between AUC-ROC and AUC-PR are higher in imbalanced situation than that in balanced case.

*3)* As the imbalancedness degree increases from 1:1 to 1:5, the ROC-AUC values of all acceptor/donor datasets increase smoothly (∼0.3% average), but the ROC-PR values decrease sharply (from 1.09% to 4.23%). It means that the number of negative samples has a greater impact on the performance of the CNN model.

*4)* DeepSS-C generates better results on the donor than the acceptor splice sites, and the reason perhaps is that the donor splice sites is more conservative than the acceptor splice sites.

Overall, DeepSS-C gives consistent performance across different datasets (from small CE donor dataset which includes 500 positive samples and 500 negative samples to large HS$^3$D donor dataset which is composed of 2880 positive samples and 14400 negative samples).

Besides AUC-ROC and AUC-PR, the values of accuracy, precision, sensitivity, specificity and MCC are also computed with 10-fold cross validation. The overall performance for balanced and imbalanced acceptor/donor splice sites prediction is shown in Fig. 3 and Fig. 4, respectively.

### C. COMPARISON WITH THE EXISTING METHODS ON HS$^3$ D DONOR DATASET

In this section we first compare the performance of DeepSS-C with five state-of-the-art sequence-based discriminative methods which identified splice sites by extracting numerical combinational features based on statistical approaches on HS$^3$D donor splice sites. These methods include Maximum Entropy Model (MEM) [49], Maximal Dependency Decomposition (MDD) [50], Weighted Matrix

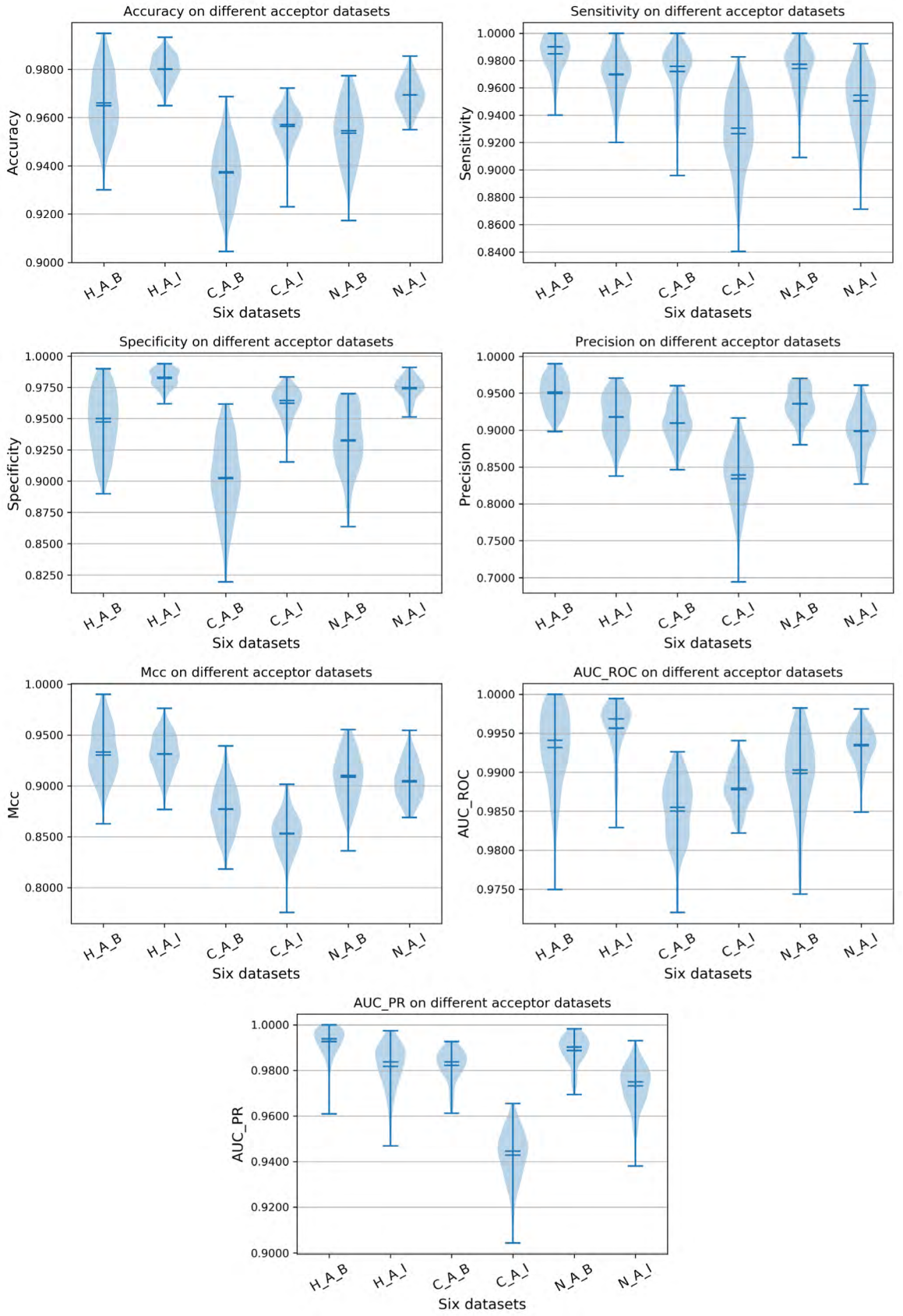



**FIGURE 3.** The overall performance on balanced/imbalanced acceptor datasets using DeepSS-C (H – HS$^3$D, C- CE, N – NN269, B - Balanced, I – Imbalanced, A - Acceptor).

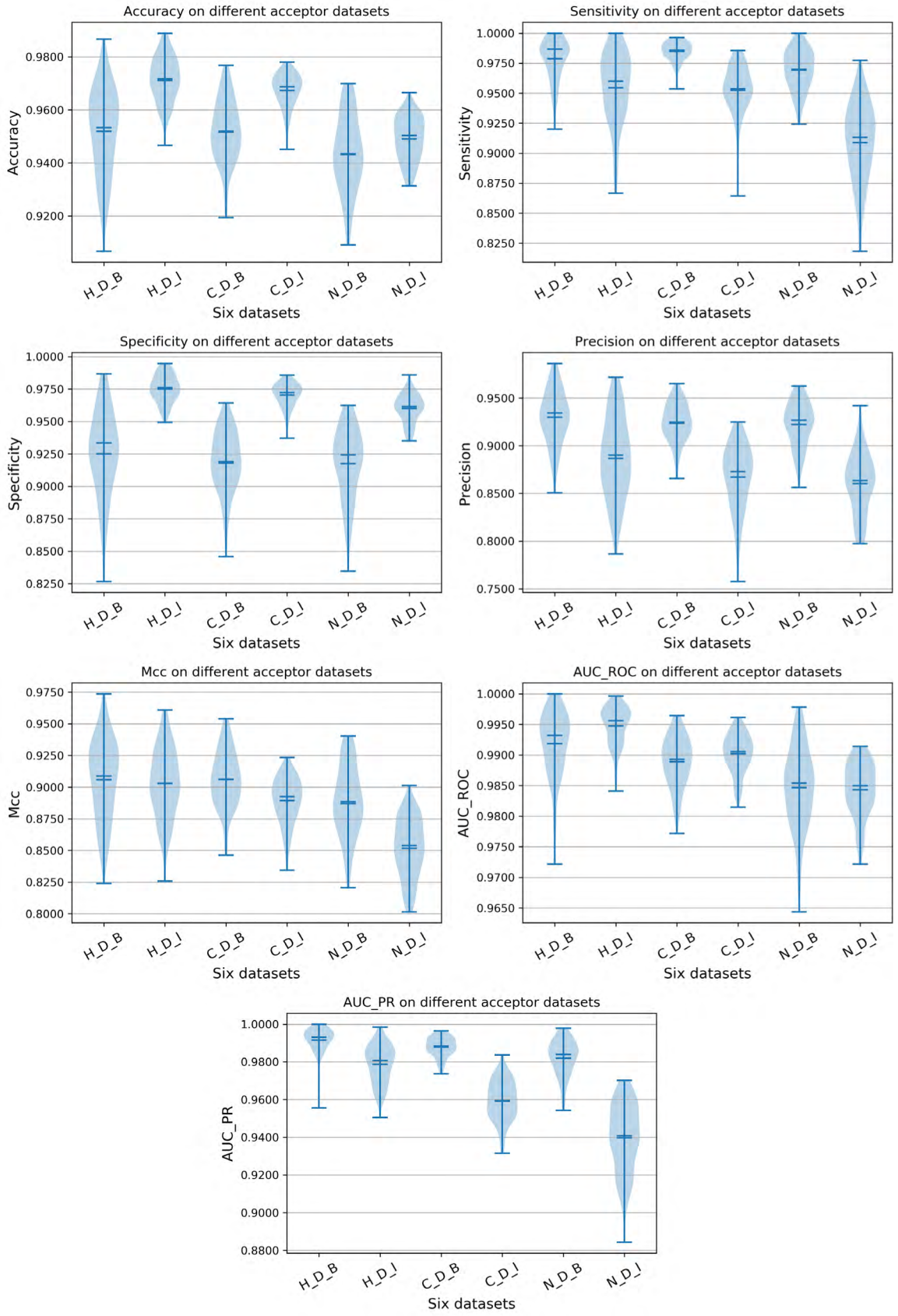



**FIGURE 4.** The overall performance on balanced/imbalanced donor datasets using DeepSS-C (H – HS$^3$D, C- CE, N – NN269, B - Balanced, I – Imbalanced, D - Donor).

**TABLE 4. Performance comparison of DeepSS-C with other existing methods on HS$^3$D donor dataset.**

| Dataset | Measure | Ratio | Methods | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | MEM [49] | MDD [54] | WMM [51] | SAE [53] | MM1 [52] | DeepSS-C |
| *HS$^3$D Donor* | *AUC-ROC* | ***1:1*** | 0.948 ± 0.0031 | 0.945 ± 0.0031 | 0.927 ± 0.0036 | 0.946 ± 0.0031 | 0.945 ± 0.0031 | **0.9889 ± 0.0041** |
| | | ***1:2.5*** | 0.946 ± 0.0031 | 0.942 ± 0.0032 | 0.924 ± 0.0036 | 0.945 ± 0.0031 | 0.941 ± 0.0032 | **0.9894 ± 0.0031** |
| | | ***1:5*** | 0.947 ± 0.0030 | 0.944 ± 0.0030 | 0.924 ± 0.0035 | 0.944 ± 0.0030 | 0.936 ± 0.0032 | **0.9901 ± 0.0028** |
| | | ***1:7.5*** | 0.947 ± 0.0030 | 0.944 ± 0.0030 | 0.925 ± 0.0034 | 0.945 ± 0.0030 | 0.941 ± 0.0031 | **0.9901 ± 0.0032** |
| | *AUC-PR* | ***1:1*** | 0.947 ± 0.0031 | 0.944 ± 0.0031 | 0.924 ± 0.0037 | 0.945 ± 0.0031 | 0.942 ± 0.0032 | **0.9879 ± 0.0052** |
| | | ***1:2.5*** | 0.878 ± 0.0045 | 0.872 ± 0.0046 | 0.867 ± 0.0046 | 0.876 ± 0.0045 | 0.870 ± 0.0046 | **0.9809 ± 0.0066** |
| | | ***1:5*** | 0.773 ± 0.0055 | 0.769 ± 0.0055 | 0.703 ± 0.0060 | 0.772 ± 0.0055 | 0.765 ± 0.0056 | **0.9678 ± 0.0087** |
| | | ***1:7.5*** | 0.683 ± 0.0059 | 0.680 ± 0.0059 | 0.675 ± 0.0060 | 0.682 ± 0.0059 | 0.679 ± 0.0060 | **0.9572 ± 0.0098** |

**TABLE 5. Performance comparison of DeepSS-C with other existing methods on HS$^3$D donor dataset.**

| HS$^3$D_Donor | Ratio | Methods | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | MM1-SVM [3] | LIK-SVM [55] | WD-SVM [56] | WDS-SVM [57] | EFFECT [20] | HSplice[10] | DeepSS-C |
| *AUC-ROC* | ***1:1*** | 97.07 | 97.13 | 97.25 | 97.06 | 97.15 | 96.05 | **98.85** |
| | ***1:5*** | 97.32 | 97.61 | 97.73 | 97.30 | 97.42 | 97.21 | **99.00** |
| *AUC-PR* | ***1:1*** | 96.78 | 97.52 | 97.67 | 97.38 | 97.58 | 97.64 | **98.73** |
| | ***1:5*** | 89.95 | 92.23 | 92.36 | 92.17 | 92.41 | 93.24 | **95.86** |

Method (WMM) [51], Markov model of first order (MM1) [52] and Sum of absolute error (SAE) method [53]. Due to different methods use different ratio of true versus false splice sites and k-fold cross validation, to be fair, we construct the same ratio of true versus false splice sites as the literature [53]. The experimental results with the same 10-fold cross validation are given in Table 4.

From Table 4, DeepSS-C outperforms all the other methods for donor splice sites prediction. AUC-ROC/AUC-PR values of DeepSS-C reach 0.9889/0.9879, 0.9894/0.9809, 0.9901/0.9678 and 0.9901/0.9572 under the ratio of 1:1, 1:2.5, 1:5 and 1:7.5, respectively. With the rising of positive and negative sample ratio, AUC-ROC increases but AUC-PR decreases. Specifically, the AUC-ROC/AUC-PR value of the worst WMM classifier are 6.19%/6.39%, 6.54%/11.39%, 6.61%/26.48% and 6.51%/28.22% lower than DeepSS-C. Even for previous the best method MEM [49], the value of AUC-ROC/AUC-PR have improved ∼4.09%/4.09%, ∼4.34%/10.29%, ∼4.31%/19.48%, ∼4.31%/27.42%, respectively, which validates that our model DeepSS-C can give an excellent performance.

In addition, DeepSS-C is also compared to SVM with MM1 encoding (MM1-SVM) [3], SVM with locally improved kernel (LIK-SVM) [42], SVM with weighted degree kernel (WD-SVM) [42], SVM with weighted degree shift kernel (WDS-SVM) [42], EFFECT [20] and HSplicer which are provided in [10]. Table 5 shows the performance comparison of these methods. To achieve a fair performance comparison between our method and these six methods, DeepSS-C is redo with 5-flod cross-validation according to original literature [10]. The comparative analysis across these methods indicates that DeepSS-C performs better than all the other methods (Table 5) in [10]. Both for balanced dataset (1:1 ratio) and imbalanced dataset (1:5 ratio), AUC-ROC and AUC-PR values of DeepSS-C (98.85%/98.73%, 99.00%/95.86%) are ranked first, ∼1.60%/∼1.06% and ∼1.27%/∼3.50% higher than the second best approach. This result also suggests that the prediction ability of DeepSS-C is higher than those of the other six methods.

From Table 4 and Table 5, it can be seen that our DeepSS-C manifested superior performance relative to the other eleven methods. Taken together, the comparative analysis demonstrates that DeepSS-C achieves comparable classification performance, in contrast to sophisticated feature extraction step.

**TABLE 6.** Performance comparison of DeepSS-C with other existing methods on HS$^3$D acceptor dataset.

| Dataset | Measure | Ratio | Methods | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | MM1-SVM [3] | MM2-SVM [58] | MCM-SVM [5] | MM1-RF [59] | MM1-RF [59] | MCM-RF [59] | DeepSS-C |
| *HS$^3$D_Acceptor* | *AUC-ROC* | ***1:1*** | 95.43 | 96.00 | 96.52 | 96.36 | 96.62 | 96.62 | **98.79** |
| | | ***1:10*** | 95.78 | 96.25 | 96.66 | 96.52 | 96.73 | 96.64 | **98.62** |

**TABLE 7.** Performance comparison between DeepSS and other 16 methods on NN269 acceptor/donor datasets.

| NN269 | Acceptor | | Donor | |
|---|---|---|---|---|
| | AUC_ROC | AUC_PR | AUC_ROC | AUC_PR |
| *IC-S-SVM [60]* | 96.28 | - | 96.66 | - |
| *MC-SVM [3]* | 96.74 | 88.33 | 97.64 | 89.57 |
| *MM1-SVM [3]* | 97.41 | - | 97.90 | - |
| *LIK-SVM [55]* | 98.19 | 92.48 | 98.04 | 92.65 |
| *WD-SVM [56]* | 98.16 | 92.53 | **98.50** | 92.86 |
| *WDS-SVM [57]* | 98.65 | 94.36 | 98.13 | 92.47 |
| *FDDM-SVM [22]* | 97.93 | 92.28 | 98.31 | 92.77 |
| *FDDM-Adaboost [22]* | 98.51 | 93.69 | 98.20 | 93.02 |
| *HSplice [10]* | - | - | 96.53 | 93.54 |
| *K-mer [61]* | 63.30 | 75.50 | 90.08 | 90.10 |
| *Gibbs Sampling [20]* | 62.80 | 72.40 | 88.80 | 90.50 |
| *EFFECT [20]* | 97.70 | 94.30 | 98.20 | 92.81 |
| *PWM [62]* | 97.10 | 90.60 | 97.70 | 91.90 |
| *BayesNetwork [20]* | 97.25 | 90.60 | 97.70 | 90.90 |
| *HomogenousHMM [20]* | 59.2 | 26.3 | 86.3 | 71.5 |
| *InHomogenousHMM [20]* | 96.78 | 88.41 | 98.18 | 92.42 |
| *DeepSS-C* | **99.05** | **96.70** | 98.40 | **93.48** |

*Note: - means not available.*

### D. COMPARISON WITH THE EXISTING METHODS ON HS$^3$D ACCEPTOR DATASET

10-fold cross validation procedure is also adopted to evaluate the performance between DeepSS-C and SVM, RF with MM1 encoding [3], second order Markov model (MM2) encoding [57], and the Markov Chain Model (MCM) [4, 5] encoding on HS$^3$D acceptor splice sites. Because the AUC-PR value in original paper is unavailable, only AUC-ROC value is adopted as evaluation criteria. Table 6 gives the detail performance comparisons with the different encoding methods. From Table 6, we can see that there have small differences in performance for different encoding methods with the same machine learning algorithm. This indicates that the existing feature extraction step cannot extract the non-linear information from the sequence completely. Similarly, our method DeepSS-C also greatly outperforms the best classifier MM1-RF [58] on HS$^3$D acceptor dataset, with increases in AUC-ROC score by $\sim$2.17% for balanced dataset and $\sim$1.89% for imbalanced dataset, respectively. The results demonstrate that CNN has the ability to capture high-level features from splice sites sequence.

### E. COMPARISON WITH THE EXISTING METHODS ON NN269 ACCEPTOR/DONOR DATASETS

In addition, to estimate the reproducibility and consistency of DeepSS-C, we also test our method on NN269 donor/acceptor datasets. The results are given in Table 7. From Table 7, it can be seen that DeepSS-C also

**TABLE 8.** The performance comparison between DeepSS-C and other existing methods on CE acceptor/donor dataset.

| CE | Acceptor | | Donor | |
|---|---|---|---|---|
| | AUC_ROC | AUC_PR | AUC_ROC | AUC_PR |
| *K-mer [61]* | *88.20* | *15.80* | *83.10* | *6.2* |
| *Gibbs Sampling [20]* | *84.20* | *80.40* | *79.10* | *80.30* |
| *EFFECT [20]* | *97.90* | *90.20* | *96.70* | *91.30* |
| *PWM [62]* | *63.60* | *7.02* | *62.50* | *4.80.* |
| *BayesNetwork [20]* | *64.20* | *6.90* | - | - |
| *HomogenousHMM [20]* | *75.03* | *12.62* | *78.30* | *13.90* |
| *InHomogenousHMM [20]* | *75.71* | *11.3* | *77.90* | *12.30* |
| *MSP [63]* | *76.80* | *13.90* | *78.21* | *13.50* |
| *WD-SVM [56]* | *99.36* | *86.7* | *99.50* | *88.20* |
| *WDS-SVM [57]* | *99.20* | *89.10* | ***99.80*** | *90.10* |
| *DeepSS-C* | ***99.64*** | ***95.88*** | *99.43* | ***92.88*** |

*Note: - means not available.*

yields the highest prediction performance on NN269 acceptor dataset, ∼0.40%/∼2.34% (AUC-ROC/AUC-PR) higher than that of all other sixteen methods. For NN269 donor dataset, the AUC-ROC value of the proposed approach is slightly worse (about ∼0.10%) than the highest WD-SVM algorithm and the AUC-PR value is 0.06% lower than HSplice [10], we guess that the reason for the decreasing in AUC-ROC value is that the sequence length in NN269 donor dataset is only 15 bases, far less than 90 bases in the NN269 acceptor dataset. For splice sites prediction, a deep CNN model first scans a DNA sequence to obtain the inter dependency of DNA sequence as low-layer motif features and then forms high-level complex features through nonlinear transformation layers. The relationship between sequence length and experimental performance is discussed in the following sections.

### F. PERFORMANCE COMPARISONS OF DEEPSS-C WITH EXISTING METHODS ON CE DATASETS

To further validate the generalizability of DeepSS-C, we check the predictive performance on CE dataset that obtained from literature [20]. In [20], the kernel-based methods have the highest overall performance (WD-SVM [55] ranks first and EFFECT [20] is the second best) both AUC-ROC and AUC-PR on the acceptor dataset, and all methods are comparable on the donor dataset. From Table 8, we can see that DeepSS-C achieves the highest overall performance (99.64%/95.88%) in terms of AUC-ROC/AUC-PR on the acceptor dataset. For donor dataset, the AUC-PR value (92.88%) of DeepSS-C ranks the first and the AUC-ROC value of DeepSS-C is just 0.37% lower than the best one. Note that, CE datasets are imbalanced and AUC-PR value can better reflect classification performance. The AUC-PR value of DeepSS-C achieves the highest on both the acceptor and donor dataset. On the acceptor dataset, DeepSS-C obtains an AUC-PR of 95.88%, followed by EFFECT [20] with a value of 90.20%. On the donor dataset, DeepSS-C obtains an AUC-PR of 92.88%, followed by EFFECT [20] with a value of 91.30 %. It indicates that our method has 5.68% AUC-PR improvement in acceptor dataset and 1.58% AUC-PR improvement in donor dataset compared with the existing best method. Even when the class distribution is heavily imbalanced, the AUC-ROC values are also higher than 99.00%, it demonstrates that DeepSS-C performs better than the other existing methods.

### G. THE CORELATIONS OF SEQUENCE LENGTH AND THE DISCRIMINATION ABILITY OF DEEPSS-C

Previous researchers do some statistical analysis based on the splice sites signal patterns, and show remarkable conservativeness within a few nucleotides near splice sites on the true splice site sequences [63]. Meanwhile, this preference is not found around the false splice site and the nucleotides on either side of the false site appear at equal probabilities. Therefore, we believe that the nucleotides of different length within the certain upstream and downstream have underlying impact on the true splice sites detection. In addition, introns also show more remarkable conservativeness than exons. Hence, we select more nucleotides in the introns than those in exons. We evaluate the correlation between sequence length and the discrimination ability of DeepSS-C model. Taking into account the shortest length of NN269 donor dataset (7 nucleotides on the left and 6 nucleotides on the right), subsequence is intercepted from the first nucleotide near true

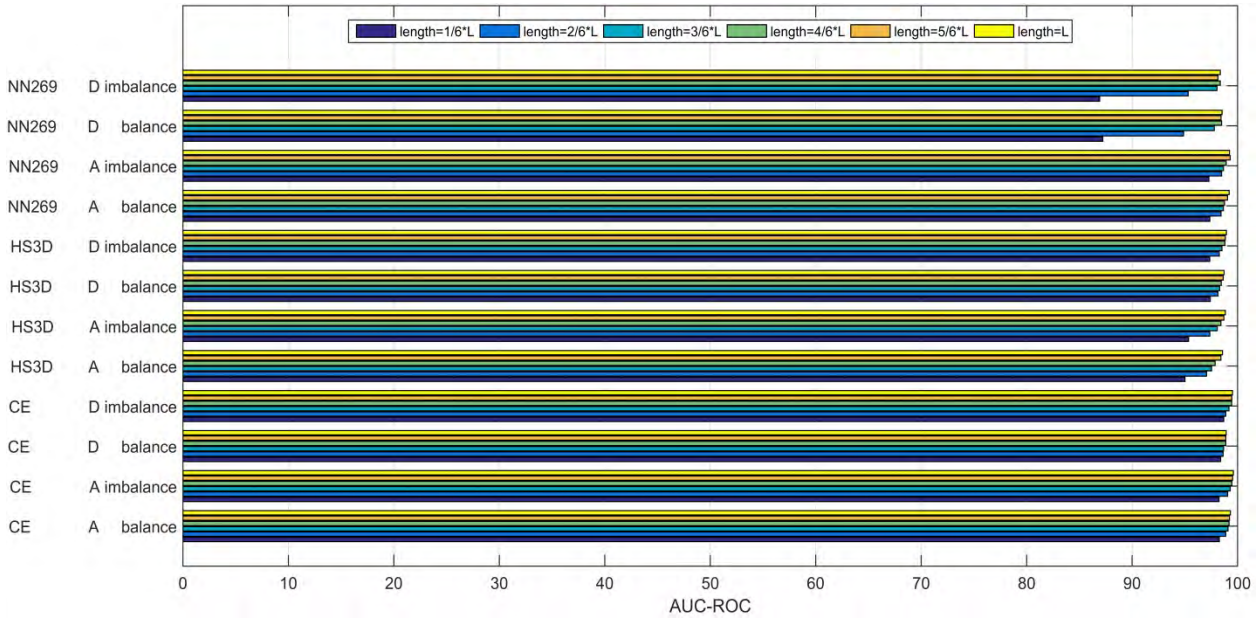



**FIGURE 5.** The effect of different sequence length on splice sites prediction with AUC-ROC (L denotes the full length of the original sequence).

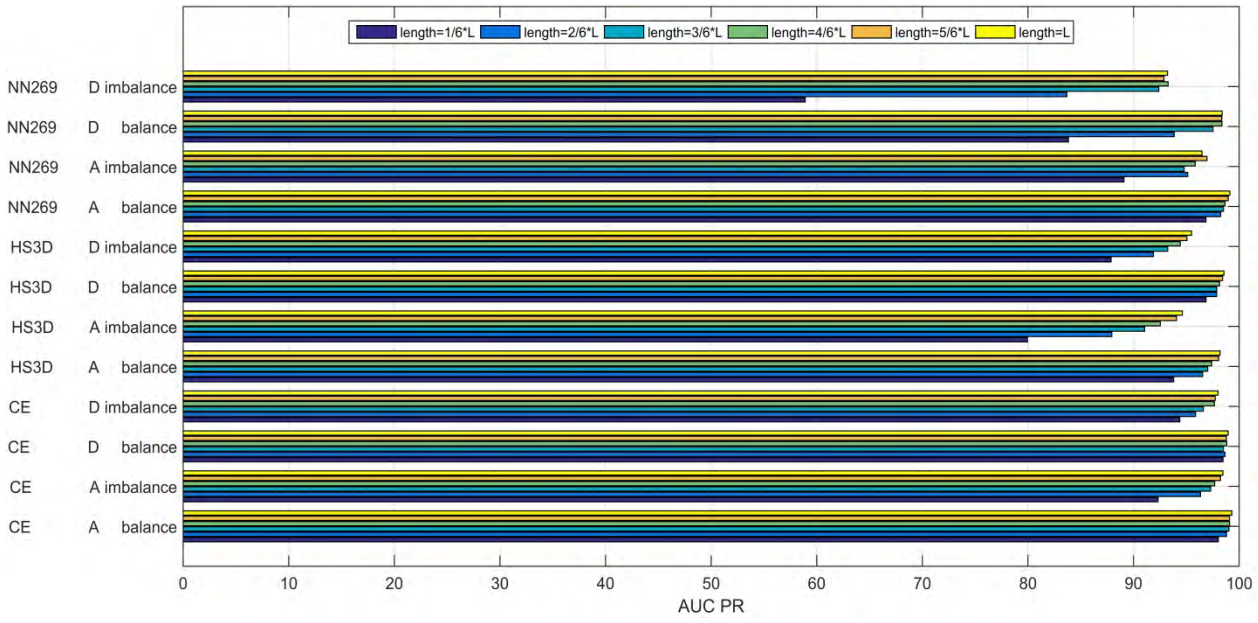



**FIGURE 6.** The effect of different sequence length on splice sites prediction with AUC-PR (L denotes the full length of the original sequence).

splice sites and extended along upstream and downstream with tolerance $d = 1/6 \times L$($L$ denotes the number of nucleotides of the whole sequence). Fig. 5 and Fig. 6 show the obvious variations between sequence length and DeepSS-C model performance on balance and imbalance datasets, respectively.

Globally, we can see that a positive correlation between the prediction performance of DeepSS-C and DNA sequence length. The values of AUC-ROC and AUC-PR on all datasets are on the rise as the length of the sequence increases. Taking CE acceptor imbalanced dataset as an example, the total sequence length is 141. When the sub-sequence length increases from 26 to 141 nucleotides, the values of AUC-ROC and AUC-PR are increased by 1.36% and 6.14%, respectively. Both of AUC-ROC and AUC-PR, the overall trend is on the rise. The increasing prediction performance of CNN is concluded to come from the non-linearity dependency underlying true splice sites and the nucleotides near them which is introduced by the longer sequence. This also explains the reason why DeepSS-C performs slightly worse

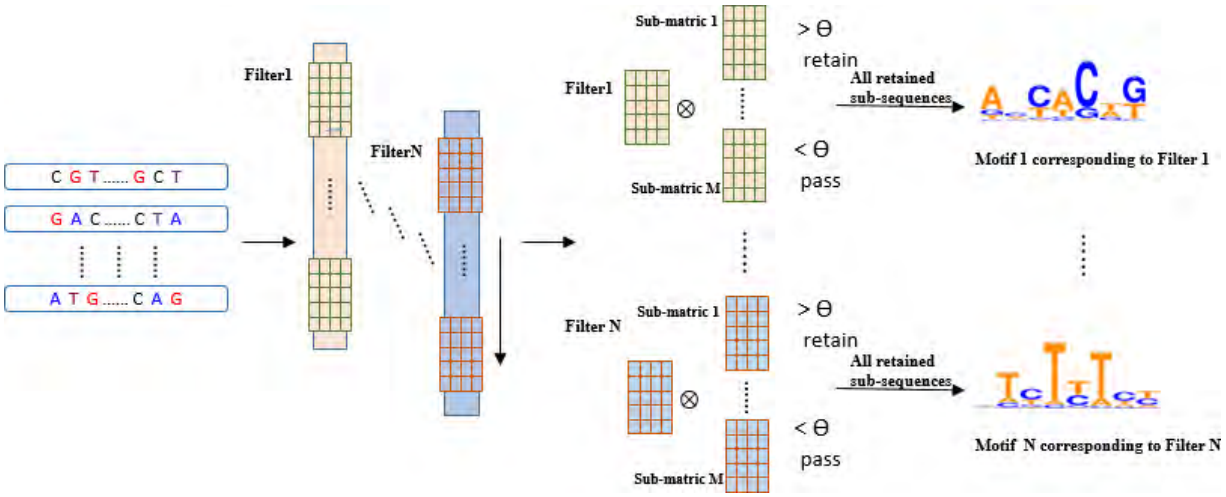



**FIGURE 7.** Motif extraction process of DeepSS-M module (from left to right).

(AUC-ROC/AUC-PR are about ∼0.10%/∼0.06% lower than the best one) on NN269 donor dataset. NN269 donor sequence length is just of 15 nucleotides. The lower performance does not represent the real ability of DeepSS-C. We believe that as long as the sequence length increases, the prediction ability of the proposed method will increase.

### H. MODEL INTERPRETABILITY BY DEEPSS-M

In recent years, machine learning methods with complex internal implementation are applied for resolving many biological sequence classification problems. Much effort has been made to gradually increase the performance of classification. However, acceptance of a computational model not only depends on rigorous verification on its classification accuracy, but also the ability of the user to understand the underlying mechanism in the asking question [64]. Unfortunately, compared with the increased accuracy of prediction algorithms, model interpretability underlying the algorithm's prediction is relatively weak. However, model interpretability enables one to understand the underlying mathematical implementation (how a mathematical model can achieve the goal of classification) and can do some downstream analysis (what genome factors hidden in biological sequences induce the related biological process happen). Due to general black-box character of machine learning methods among most computational models, the complex learned decision rules behind the model are very hard to interpret and thereby cannot be related to biological facts easily.

In this paper, we tackle the model interpretability challenge by extracting motifs from the first layer of DeepSS-M module via filters from a different perspective. Fig. 7 illustrates our approach. Subsequently, we demonstrate the efficiency and efficacy of our approach on $HS^3D$, NN269 and CE datasets, evaluated by means of the JASPAR 2018 database (available at http://jaspar.genereg.net) [65].

#### 1) MOTIF ANALYSIS

To interpret the reason that why the CNN architecture can greatly outperform the state-of-the-art computational approaches for splice sites prediction, the DNA sequence patterns and motifs underlying model training are explored. In this section, to make CNN-based sequence classifiers more accessible and profitable, we introduce the concept of motif, extract the motifs which promote splice sites prediction and analyze the component of motif. Motif analysis presented in the following is performed using DeepSS-M module on $HS^3D$ acceptor dataset. All motifs discovered for $HS^3D$, NN269 and CE donor/acceptor datasets are provided at http://ailab.ahu.edu.cn:8087/DeepSS/index.html.

##### *a: MOTIF DEFINITION*

A motif is referred to a set of subsequences that associated with a certain functional decision mechanism. In our splice sites experiment, a motif is the representation of any important positional k-mer subsequence fragments which regulate splice site decision mechanism. A motif can be converted approximately to the count or frequency matrices of position-specific subsequences.

##### *b: MOTIF EXTRACTION AND VISUALIZATION*

To discern underlying motifs that are positive correlation with being a true splice site, we trained the second CNN module DeepSS-M on $HS^3D$ acceptor dataset. This module has the same architecture as described in the previous section (see Methods). The only difference is that DeepSS-M is fed on all sequences from the $HS^3D$ dataset through the convolutional and rectification stages. Given the trained CNN, the core idea is to determine a motif by an according position weight matrix (PWM) metric from these subsequence fragments which maximally activated by filters of the first convolutional layer. Concretely, each convolutional filter of length $L$ is matched against all possible

one-hot encoded input sub-matrices with same shape $4 \times L$ at every position to recognize possible local subsequence feature. Equation (2) is adopted as a measure for the contribution of the positional subsequence to exercise a prediction function because a high value computed by equation (2) implies a strong contribution on being a true splice site. A subsequence fragment is retained if the $ReLU = \max(0, x)$ activation value of the subsequence at a certain position passes a threshold over all subsequences and positions. In our application, the threshold is set to half of the maximum subsequence activation value. After that, all the extracted subsequences are stacked together and the nucleotide frequencies are counted to compute a PWM. PWMs are a powerful generalization of sequence logos because they can capture and visualize sequence patterns that are relevant for the investigated biological phenomena. Here, sequence logos are formed by using WebLogo version 3.6 [66] (available at http://weblogo.threeplusone.com/create.cgi). After the above steps, a motif approximately corresponding to the PWM is visualized. Fig.8 (a) shows a motif 'GGGCAGGG' detected by filter 98 of DeepSS-M model on HS$^3$D acceptor dataset. In Fig. 8 (a), the size of a letter in the motif indicates the occurrence probability of the corresponding nucleotide at a certain position. Furthermore, all motifs logos discovered on HS$^3$D, NN269 and CE datasets are available at http://ailab.ahu.edu.cn:8087/DeepSS/index.html.

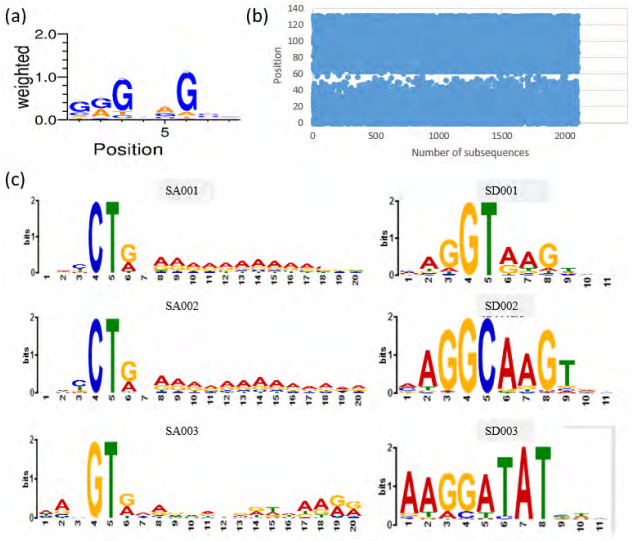



**FIGURE 8.** Motif example, subsequence distribution and identified motifs by DeepSS-M. (a) a sequence logo of a motif filter 98 and corresponding matched known motifs for this filter are shown in (c). All position of subsequences according to the motif detected by filter 98 is illustrated in (b).

### *c: SUBSEQUENCES DISTRIBUTION*

A motif is derived from a set of maximally activated subsequences filtered from the input sequence. Besides of the most relevant motifs extracted by DeepSS-M, the corresponding starting positions of these detected subsequences are also counted. Fig.8(b) shows the filtered subsequences position distribution corresponding to filter 98. HS$^3$D acceptor dataset [41] sequences are 140 nucleotides and conserved nucleotides AG lie at 69th and 70th positions. Within the scope of 140bp, the first discriminative subsequence starts at position 0 and consists of 8 nucleotides. A striking drop appearing from position 40 to 63 indicates that the activations of subsequences in this scope are lower than the other positions in the sequence. There is a straight line at position 69, which happens to the true splice site 'AG' appearance position. It demonstrates that subsequences around true splice sites retain with a probability of 100 percent and have more important influence in deciding a site true or false. Furthermore, subsequences distribution density in upper half appears significantly higher than lower half. Altogether, the Figure indicates a general fact that there exist more influential subsequences in the right part of splice sites sequence than left part, from which we conclude that the discriminative subsequences in the right part is more conservative than the left part.

### 2) MOTIF VALIDATION

To validate the motifs generated by DeepSS-M on HS$^3$D acceptor dataset, JASPAR2018 database [65] is chosen. It provides us with a collection of important DNA motifs. JASPAR SPLICE score [67] is adopted as a measure of the motif reconstruction quality (MRQ). Motifs discovered by DeepSS-M are matched to annotated motifs in the JASPAR2018 database [65] with the JASPAR splice, using Tomtom (Version 4.12.0) from the MEME-Suite [68]. The motif comparison function is set to Pearson correlation coefficient and E-value $< 10$.

Take filter 98 as an example, the corresponding detected motif matches to six existing motifs ("SA0001", "SA0002", "SA0003", "SD0002", "SD0001" and "SD0003") in the JASPAR2018 database [65]. "SXXXX" is the ID of the matched motif. Except for motif "SD0003", the overlap (the number of letters that overlapped in the optimal alignment) values of other five motifs are 8, which are equal to filter length in DeepSS-M and thereby demonstrate that the motifs of DeepSS-M detected are useful. The matched motifs in the JASPAR2018 database [65] are shown in Fig. 8 (c).

### 3) FILTER ANALYSIS

Besides motif analysis, we also investigate the ability of learnt convolve filters in the first convolutional layer by DeepSS-M. The role of a filter acts as a motif detector. It can learn particularly splice site decision pattern and identify candidate motifs from the local sequence context.

#### *a: FILTER VISUALIZATION*

To visualize a $m \times 4$ filter (m is the length of filter), heatmap as indicated by light blue, yellow, orange, or red colors (as seen in Fig. 9 (a), length $L = 8$, color represents the weight of one certain nucleotides) is adopted to show the nucleotide component proportion at each position using

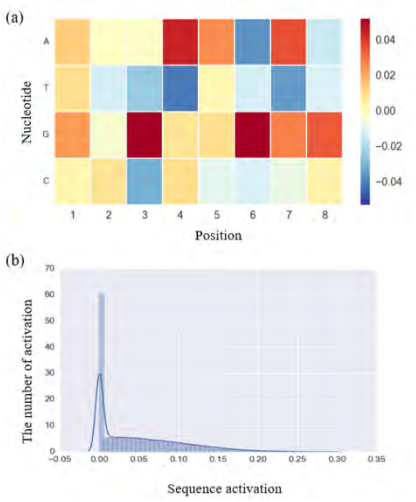



**FIGURE 9.** The heatmap of learned weights of convolve filters of CNN and density map are shown in (a) and (b), respectively.

learnt filter weight matrix. Learned weights of convolve filter in DeepSS-M can be represented as a sequence logo (for example "GCGAAGGG"). Each letter in the logo denotes an extraordinary high score according to each specific position. Before display, we manipulate the coefficients by subtracting the mean from each column (each motif position) of the matrix. The visualization of result is more easily interpreted.

### *b: FILTER ABILITY*

Filter activity (the motif occurrence frequency in sequence windows) can reflect the ability of a filter in capturing the truly relevant motifs. The value of filter activity for a set of sequences is quantified as the average of mean sequence activations. Specially, we first mean all subsequences activation by overlapping each $m \times 4$ filter on $n - m + 1$ ($n$ denotes the length of the sequence, $m$ is the filter length) subsequences for one DNA sequence and then take the average of all subsequences mean activation as the final activity value. The higher the filter activity value, the more ability the filter has and the larger influence the motif have on determining splice site as a true one. Variance of each filter activity that reflects the fluctuation range of the activations for all sequences is also calculated. Additionally, filter density of the first convolutional layer for DeepSS-M model on $HS^3D$ balance acceptor dataset are computed. Density map displays the occurrence distribution of all activations according to each filter on all DNA sequences.

Top twenty activities are shown in Table 9 and the first filter activity density is illustrated in Fig. 9 (b). The highest activity value of the 98th filter on HS3D balance acceptor dataset is around 0. 0376. From the overall perspective, mean and variance of filter activity decreases simultaneously. Activity density illustration (Fig. 9 (b)) for filter 98 shows that the whole subsequences activation values corresponding to the 98th filter are less than 0.30 and the majority value is close to 0.05, which is consistent to the mean/variance activation for filter 98 in Table 9.

**TABLE 9.** Mean/variance activation for top twenty filters of the first convolutional layer derived from DeepSS-M.

| Order | Filter ID | Filter | Mean | Variance | Number of subsequences |
|---|---|---|---|---|---|
| *1* | 98 | GCGAAGGG | 0.0376 | 0.0489 | 22829 |
| *2* | 5 | AATTGCGC | 0.0342 | 0.0429 | 22863 |
| *3* | 19 | AGGGCGGC | 0.0333 | 0.0412 | 17272 |
| *4* | 74 | TGCGAAAG | 0.0330 | 0.0378 | 26938 |
| *5* | 68 | AGATGTCG | 0.0328 | 0.0466 | 25494 |
| *6* | 45 | GCTGGGGG | 0.0321 | 0.0451 | 17544 |
| *7* | 84 | GAGTGAGC | 0. 0317 | 0. 0464 | 21676 |
| *8* | 15 | GGAGGATA | 0.0306 | 0. 0417 | 16732 |
| *9* | 22 | TTGGGCCG | 0.0300 | 0. 0415 | 21296 |
| *10* | 69 | CGGCGCGA | 0.0298 | 0. 0419 | 18661 |
| *11* | 23 | ATTGAAGA | 0.0295 | 0. 0435 | 14265 |
| *12* | 126 | GGGACGAT | 0.0290 | 0.0419 | 22904 |
| *13* | 135 | TATCCTAG | 0.0286 | 0.0383 | 27238 |
| *14* | 65 | CTGAGACA | 0.0283 | 0.0451 | 17391 |
| *15* | 83 | GAGGAGAG | 0.0280 | 0.0415 | 21248 |
| *16* | 34 | AGCTGCGG | 0.0277 | 0.0434 | 13111 |
| *17* | 21 | ATAGAATA | 0.0276 | 0.0355 | 22755 |
| *18* | 31 | CACGAACG | 0.0275 | 0.0423 | 10705 |
| *19* | 124 | CATTCAGC | 0.0273 | 0.0384 | 23873 |
| *20* | 52 | TACAACAT | 0.0271 | 0.0383 | 21631 |

### 4) FILTER CO-OCCURRENCE AND MOTIF SUMMARY STATISTICS

We apply principal component analysis on the mean subsequence activations to quantify filters co-occurrence and motifs composition that associated with splice sites pattern across sequence windows. The first two principal components of the filter activity are adopted to cluster the 141 filters of DeepSS-M. Since a filter corresponds to a motif, we investigate filter co-occurrence and motif composition simultaneously (illustrated in Fig. 10). The color denotes the estimated motif effect associated with splice site and subsequence logos are shown for representative motifs. We can observe that the value of filters activities lies in 0.0140 and 0.0200 is more intensive relatively to other places in the Figure (left bottom part in Fig. 10). As a result, we can infer that motifs with similar nucleotide composition tend to co-occur around splice site and motifs associated with splice sites tend to be CT-rich.

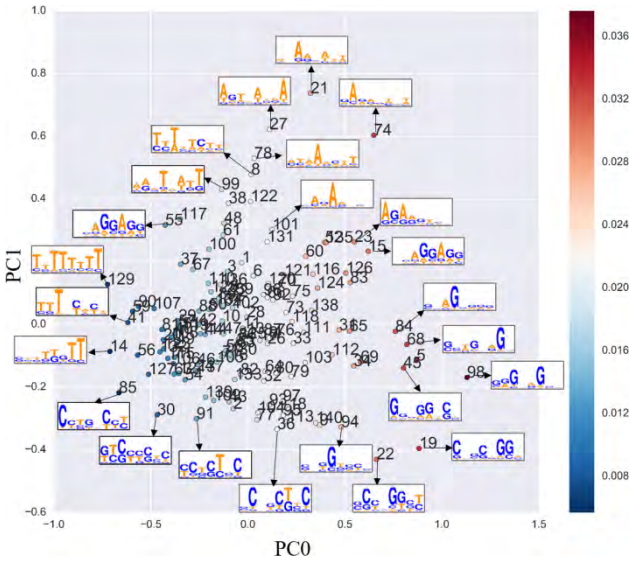



**FIGURE 10.** Co-occurrence between filters and motifs by PCA. Discovered splice sites-associated sequence motifs. The estimated motif effect on splice sites is shown by color. Sequence logos are shown for representative motifs.

## IV. CONCLUSION

The prediction of splice sites has long been considered an important task in the studies of gene expression regulation. Unlike previous methods, we propose a deep CNN framework, namely DeepSS, which consists of DeepSS-C module for splice sites prediction and DeepSS-M module for model interpretability, respectively. DeepSS-C module solely depends on sequence characteristics to maximize the discrimination between splice sites and non-splice sites. It can capture nonlinear dependencies and interaction effects from local DNA sequence windows and span wider sequence context at multiple genomic scales to predict splice site. A major feature of DeepSS-C is its convolutional architecture, which consists of just two convolutional and two pooling layer to detect informative sequence motifs within and across the local sequence context and two fully connected layers for modeling motif interactions. The steady performance demonstrates that DeepSS-C is really a very promising predictor for splice sites prediction. Additionally, there is a positive correlation between the prediction ability of DeepSS-C and DNA sequence length. In the subsequently model interpretability section, the parameters of the trained DeepSS-M module is used for model interpretability and downstream analysis, including: i) genome factors detection (the truly relevant motifs); ii) the ability of CNN filters on detecting motifs; iii) co-analysis of filters and motifs on DNA sequence pattern. We believe that the developed CNN framework DeepSS is capable of grasping high-level features, identifying potential splice sites, extracting the truly relevant motifs, and discovering splice site-associated sequence pattern, which will provide useful insights into understanding the mechanisms of gene expression.

## ACKNOWLEDGMENT

The authors would like to thank editors and reviewers for careful reading of the manuscript. The authors would also like to thank the anonymous reviewers for their valuable comments and suggestions, which were helpful for improving the quality of the paper.

## REFERENCES

[1] S. Sonnenburg, G. Rätsch, A. Jagota, and K.-R. Müller, "New methods for splice site recognition," in *Proc. Int. Conf. Artif. Neural Netw.*, vol. 2415. Berlin, Germany: Springer, 2002, pp. 329–336.
[2] A. Malousi, I. Chouvarda, V. Koutkias, S. Kouidou, and N. Maglaveras, "SpliceIT: A hybrid method for splice signal identification based on probabilistic and biological inference," *J. Biomed. Inform.*, vol. 43, no. 2, pp. 208–217, 2010.
[3] A. Baten, B. Chang, S. K. Halgamuge, and J. Li, "Splice site identification using probabilistic parameters and SVM classification," *BMC Bioinf.*, vol. 7, p. S15, Dec. 2006.
[4] L. S. Ho and J. C. Rajapakse, "Splice site detection with a higher-order Markov model implemented on a neural network," *Genome Informat.*, vol. 14, no. 14, pp. 64–72, 2003.
[5] N. Goel, S. Singh, and T. C. Aseri, "An improved method for splice site prediction in DNA sequences using support vector machines," *Procedia Comput. Sci.*, vol. 57, pp. 358–367, Aug. 2015.
[6] D. Wei, H. Zhang, Y. Wei, and Q. Jiang, "A novel splice site prediction method using support vector machine," *J. Comput. Inf. Syst.*, vol. 9, no. 20, pp. 8053–8060, 2013.
[7] J. Huang, T. Li, K. Chen, and J. Wu, "An approach of encoding for prediction of splice sites using SVM," *Biochimie*, vol. 88, no. 7, pp. 923–929, 2006.
[8] X. Zhang, J. Lee, and L. A. Chasin, "The effect of nonsense codons on splicing: A genomic analysis," *RNA*, vol. 9, no. 6, pp. 637–639, 2003.
[9] A. T. M. G. Bari, M. R. Reaz, and B.-S. Jeong, "Effective DNA encoding for splice site prediction using SVM," *Match Commun. Math. Comput. Chem.*, vol. 71, no. 1, pp. 241–258, 2013.
[10] P. K. Meher, T. K. Sahu, A. R. Rao, and S. D. Wahi, "Identification of donor splice sites using support vector machine: A computational approach based on positional, compositional and dependency features," *Algorithms Mol. Biol.*, vol. 11, no. 1, p. 16, 2016.
[11] P. K. Meher, T. K. Sahu, and A. R. Rao, "Prediction of donor splice sites using random forest with a new sequence encoding approach," *BioData Mining*, vol. 9, no. 1, p. 4, 2016.
[12] E. Pashaei, M. Ozen, and N. Aydin, "Splice sites prediction of human genome using AdaBoost," in *Proc. IEEE-EMBS Int. Conf. Biomed. Health Inform.*, Feb. 2016, pp. 300–303.
[13] Y. Bengio, A. Courville, and P. Vincent, "Unsupervised feature learning and deep learning: A review and new perspectives," *CoRR*, vol. 1, pp. 1–30, Jun. 2012.
[14] B. Draper and J. C. B. Filho, "Feature selection from huge feature sets in the context of computer vision," Colorado State Univ., Fort Collins, CO, USA, Tech. Rep. 2, 2000, p. 159.
[15] J. Neumann, C. Schnörr, and G. Steidl, *Combined SVM-Based Feature Selection and Classification*. Norwell, MA, USA: Kluwer, 2005, pp. 129–150.
[16] G. Dror, R. Sorek, and R. Shamir, "Accurate identification of alternatively spliced exons using support vector machine," *Bioinformatics*, vol. 21, no. 7, pp. 897–901, 2005.
[17] Y. Zhang, C.-H. Chu, Y. Chen, H. Zha, and X. Ji, "Splice site prediction using support vector machines with a Bayes kernel," *Expert Syst. Appl.*, vol. 30, no. 1, pp. 73–81, 2006.
[18] D. Wei, W. Zhuang, Q. Jiang, and Y. Wei, "A new classification method for human gene splice site prediction," in *Health Information Science*. Berlin, Germany: Springer, 2012.
[19] H. S. Lopes, C. R. E. Lima, and N. J. Murata, "A configware approach for high-speed parallel analysis of genomic data," *J. Circuits Syst. Comput.*, vol. 16, no. 04, pp. 527–540, 2007.
[20] U. Kamath, J. K. De, and A. Shehu, "Effective automated feature construction and selection for classification of biological sequences," *PLoS ONE*, vol. 9, no. 7, p. e99982, 2014.

[21] Q. Zhang, Q. Peng, Q. Zhang, Y. Yan, K. Li, and J. Li, "Splice sites prediction of human genome using length-variable Markov model and feature selection," *Expert Syst. Appl.*, vol. 37, no. 4, pp. 2771–2782, 2010.

[22] E. Pashaei, A. Yilmaz, M. Ozen, and N. Aydin, "Prediction of splice site using AdaBoost with a new sequence encoding approach," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2016, pp. 3853–3858.

[23] C. Angermueller, T. Pärnamaa, L. Parts, and O. Stegle, "Deep learning for computational biology," *Mol. Syst. Biol.*, vol. 12, no. 7, p. 878, Jul. 2016.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2012.

[25] D. C. Cireşan, U. Meier, and J. Schmidhuber, "Transfer learning for Latin and Chinese characters with deep neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 20, Jun. 2012, pp. 1–6.

[26] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 4, 2014, pp. 3104–3112.

[27] L. Deng and R. Togneri, "Deep dynamic models for learning hidden representations of speech features," in *Speech and Audio Processing for Coding, Enhancement and Recognition*. New York, NY, USA: Springer, 2015, pp. 153–195.

[28] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, vol. 8689, 2014, pp. 818–833.

[29] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning," *Nature Biotechnol.*, vol. 33, no. 8, pp. 831–838, 2015.

[30] D. R. Kelley, J. Snoek, and J. L. Rinn, "Basset: Learning the regulatory code of the accessible genome with deep convolutional neural networks," *Genome Res.*, vol. 26, no. 7, pp. 990–999, 2016.

[31] J. Zhou and O. G. Troyanskaya, "Predicting effects of noncoding variants with deep learning-based sequence model," *Nature Methods*, vol. 12, no. 10, pp. 931–934, 2015.

[32] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzell, "Learning to diagnose with LSTM recurrent neural networks," in *Proc. Int. Conf. Learn. Represent.*, San Juan, Puerto Rico, 2016, pp. 1–18.

[33] R. Salakhutdinov and H. Larochelle, "Efficient learning of deep Boltzmann machines," *Proc. Mach. Learn. Res.*, vol. 9, no. 8, pp. 693–700, 2010.

[34] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," *Momentum*, vol. 9, no. 1, pp. 599–619, 2010.

[35] G. Alain, Y. Bengio, and S. Rifai, "Regularized auto-encoders estimate local statistics," *CoRR*, vol. abs/1211.4246, pp. 1–17, 2012.

[36] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[37] X. Min, N. Chen, T. Chen, and R. Jiang, "DeepEnhancer: Predicting enhancers by convolutional neural networks," in *Proc. IEEE Int. Conf. Bioinf. Biomed. (BIBM)*, Dec. 2016, pp. 637–644.

[38] R. K. Umarov and V. V. Solovyev, "Recognition of prokaryotic and eukaryotic promoters using convolutional deep learning neural networks," *PLoS ONE*, vol. 12, no. 2, p. e0171410, 2017.

[39] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[40] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[41] P. Pollastro and S. Rampone, "HS$^3$D, a dataset of homo sapiens splice regions, and its extraction procedure from a major public database," *Int. J. Mod. Phys. C*, vol. 13, no. 8, pp. 1105–1117, 2002.

[42] S. Sonnenburg, G. Schweikert, P. Philips, J. Behr, and G. Rätsch, "Accurate splice site prediction using support vector machines," *BMC Bioinf.*, vol. 8, no. 10, 2007, Art. no. 13386.

[43] M. G. Reese, F. H. Eeckman, D. Kulp, and D. Haussler, "Improved splice site detection in Genie," *J. Comput. Biol.*, vol. 4, no. 3, pp. 311–323, 1997.

[44] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfittin," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Jun. 2014.

[45] F. Chollet. (2015). Keras: Deep Learning Library for Theano and Tensorflow. GitHub Repositiry. [Online]. Available: https://github.com/fchollet/keras

[46] F. Bastien *et al.*, "Theano: New features and speed improvements," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.* Lake Tahoe, NV, USA: Curran Associates, Inc., 2012, pp. 1-10.

[47] Theano Development Team *et al.* (2016). "Theano: A Python framework for fast computation of mathematical expressions." [Online]. Available: https://arxiv.org/abs/1605.02688

[48] J. Henderson, S. Salzberg, and K. H. Fasman, "Finding genes in DNA with a hidden Markov model," *J. Comput. Biol.*, vol. 4, no. 2, pp. 127–141, 1997.

[49] G. Yeo and C. B. Burge, "Maximum entropy modeling of short sequence motifs with applications to RNA splicing signals," *J. Comput. Biol.*, vol. 11, nos. 2–3, pp. 377–394, 2004.

[50] C. Burge and S. Karlin, "Prediction of complete gene structures in human genomic DNA," *J. Mol. Biol.*, vol. 268, no. 1, pp. 78–94, 1997.

[51] R. Staden, "Computer methods to locate signals in nucleic acid sequences," *Nucl. Acids Res.*, vol. 12, pp. 505–519, Jan. 1984.

[52] N. Sheth, X. Roca, M. L. Hastings, T. Roeder, A. R. Krainer, and R. Sachidanandam, "Comprehensive splice-site analysis using comparative genomics," *Nucl. Acids Res.*, vol. 34, no. 14, pp. 3955–3967, 2006.

[53] P. K. Meher, T. K. Sahu, A. R. Rao, and S. D. Wahi, "A statistical approach for 5' splice site prediction using short sequence motifs and without encoding sequence data," *BMC Bioinf.*, vol. 15, no. 1, p. 362, 2014.

[54] A. Zien, G. Rätsch, S. Mika, B. Schölkopf, T. Lengauer, and K.-R. Müller, "Engineering support vector machine kernels that recognize translation initiation sites," *Bioinformatics*, vol. 16, no. 9, pp. 799–807, 2000.

[55] G. Rätsch and S. Sonnenburg, "Accurate splice site detection for caenorhabditis elegans," in *Kernel Methods in Computational Biology*. Cambridge, MA, USA: MIT Press, 2004, p. 277.

[56] G. Rätsch, S. Sonnenburg, and B. Schölkopf, "RASE: Recognition of alternatively spliced exons in *C.elegans*," *Bioinformatics*, vol. 21, pp. i369–i377, Jun. 2005.

[57] S. Maji and D. Garg, "Hybrid approach using SVM and MM2 in splice site junction identification," *Current Bioinf.*, vol. 9, no. 1, pp. 76–85, 2014.

[58] E. Pashaei, M. Ozen, and N. Aydin, "Splice site identification in human genome using random forest," *Health Technol.*, vol. 7, no. 1, pp. 141–152, 2017.

[59] A. Baten, S. K. Halgamuge, and B. Chang, "Fast splice site detection using information content and feature reduction," *BMC Bioinf.*, vol. 9, no. 12, p. S8, 2008.

[60] C. Leslie, E. Eskin, and W. S. Noble, "The spectrum kernel: A string kernel for SVM protein classification," in *Proc. Pacific Symp. Biocomput.*, vol. 7, 2001, pp. 564–575.

[61] N. I. Gershenzon, G. D. Stormo, and I. P. Ioshikhes, "Computational technique for improvement of the position-weight matrices for the DNA/protein binding sites," *Nucl. Acids Res.*, vol. 33, no. 7, pp. 2290–2301, 2005.

[62] J. Grau, J. Keilwagen, A. Gohr, B. Haldemann, S. Posch, and I. Grosse, "Jstacs: A java framework for statistical analysis and classification of biological sequences," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 1967–1971, 2012.

[63] M. M. Yin and J. T. L. Wang, "Effective hidden Markov models for detecting splicing junction sites in DNA sequences," *Inf. Sci.*, vol. 139, nos. 1–2, pp. 139–163, 2001.

[64] K. Hansen, D. Baehrens, T. Schroeter, M. Rupp, and K.-R. Müller, "Visual interpretation of kernel-based prediction models," *Mol. Inform.*, vol. 30, no. 9, pp. 817–826, 2011.

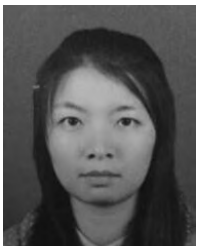
[65] A. Sandelin, W. Alkema, P. Engström, W. W. Wasserman, and B. Lenhard, "JASPAR: An open-access database for eukaryotic transcription factor binding profiles," *Nucl. Acids Res.*, vol. 32, pp. D91–D94, Jan. 2004.

[66] G. E. Crooks, G. Hon, J.-M. Chandonia, and S. E. Brenner, "WebLogo: A sequence logo generator," *Genome Res.*, vol. 14, no. 6, pp. 1188–1190, 2004.

[67] A. Sandelin, A. Höglund, B. Lenhard, and W. W. Wasserman, "Integrated analysis of yeast regulatory sequences for biologically linked clusters of genes," *Funct. Integrative Genomics*, vol. 3, no. 3, pp. 125–134, 2003.

[68] T. L. Bailey *et al.*, "MEME Suite: Tools for motif discovery and searching," *Nucl. Acids Res.*, vol. 37, pp. W202–W208, Jul. 2009.

**XIUQUAN DU** received the Ph.D. degree in computer science from Anhui University, Hefei, China, in 2010. He is currently an Associate Professor with Anhui University. His research interests include bioinformatics, medical image analysis, and machine learning.



**YU YAO** is currently pursuing the M.S. degree with the School of Computer Science and Technology, Anhui University, Hefei, China. Her research interests include bioinformatics and machine learning.

**YANYU DIAO** is currently pursuing the M.S. degree with the School of Computer Science and Technology, Anhui University, Hefei, China. Her research interests include bioinformatics and machine learning.

**HUAIXU ZHU** is currently pursuing the M.S. degree with the School of Computer Science and Technology, Anhui University, Hefei, China. His research interests include bioinformatics and machine learning.

**YANPING ZHANG** received the Ph.D. degree in computer science from Anhui University, Hefei, China, in 2003. She has been a Professor with Anhui University since 2004. Her research interests include quotient space, granular computing, and machine learning.

**SHUO LI** received the Ph.D. degree in computer science from Concordia University, Montreal, QC, Canada, in 2006. He is an Associate Professor with Western University and an Adjunct Scientist with the Lawson Health Research Institute. He is currently leading the Digital Imaging Group of London as the Scientific Director. His current research interests include automated medial image analysis and visualization and machine learning.