2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI) | 978-1-6654-0898-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICTAI52525.2021.00032

An Interpretation of Convolutional Neural Networks for Motif Finding from the View of Probability

Weinan Wang[1,†], Yuhang Guo[1,†], Wei Ju[2], Xiao Luo[1], Minghua Deng[1*]
[1]*School of Mathematical Sciences, Peking University*
[2]*Department of Computer Science and Technology, Peking University*
{*wangweinan,yuhangguo,juwei,xiaoluo,dengmh*}*@pku.edu.cn*

***Abstract*—The powerful learning ability of a convolutional neural network to perform functional classification provides valuable clues for the discovery of biological problems and image recognition. However, the mechanism of deep learning still lacks satisfying interpretation and the convolutional neural network is usually regarded as a black-box model. Consequently, it is challenging to design deep learning models and adjust the hyper-parameters for specific tasks without theoretical guidance. We raise a new task of deep learning – 2-dimensional logo detecting for computer vision and develop a novel model of convolutional neural networks to solve the task. Furthermore, we interpret this specific model from the point of view of statistical learning. With the guidance of statistical interpretation, we can design the model structure more efficiently and obtain better performances.**

***Index Terms*—Convolutional neural network, Motif finding, Interpretation, Probabilistic model**

I. Introduction

Recently, the application of convolutional neural networks has achieved great success in the field of image recognition [1]–[3]. Many models with deep convolutional neural networks have been proposed such as Deep Belief Network, [4], LSTM [5], Alexnet [6], ResNet [7]. Because deep learning is an effective method to solve supervised classification problems, DeepBind [8] and DeepSEA [9] utilized convolutional neural networks to predict RNA- and DNA-protein binding and achieved encouraging results. As a great number of training sets are drawn from high-throughput data, models of neural networks can be trained without overfitting, which can get accurate test results compared to traditional machine learning methods such as gkm-SVM [10].

The convolutional neural networks used by DeepBind [8] and DeepSEA [9] are similar to deep learning approaches in computer vision [6], [11]. Recently, many methods in deep learning such as data augmentation have been applied to predict DNA-protein binding [12]–[17] and get some results. The major change of convolutional neural networks from computer vision to genomics can be accomplished by considering a window of genome sequences as an image. Each sequence is transformed into a one-hot format. Specifically, it's turned into $4 \times L$ matrix where each base pair in the sequence is denoted as one of for one-hot vectors $[1,0,0,0]$, $[0,1,0,0]$, $[0,0,1,0]$ and $[0,0,0,1]$ and then 1-D convolutional neural networks can be utilized. In this way, the genomic task of modeling DNA sequence protein binding specificity is analogous to the computer vision task of two-class image classification where we can utilize deep learning to help increase accuracy.

One of the main difficulties of deep learning is it lacks interpretation (i.e. black-box systems [18]), which corresponds to the choice of 'gold standard' [19]. Many attempts have been done in specific networks [20], [21]. Opening the black box of deep learning is meaningful for searching for better models of neural networks.

In this paper, firstly, we promote the task of motif finding to two-dimensional images and show our tasks are meaningful for computer vision by defining logo images and motif finding the task of those images. Besides, because of the connection between DeepBind and the probabilistic model, we design a model of neural networks to deal with the task of 2-D motif finding and attempt to open up the black box of the neural network. We utilize many classic probabilistic views to interpret our models, such as hypothesis testing, Bayesian view, and Gibbs sampling [22]. We give our model a simple probabilistic interpretation and in this way, we can apply many statistical views to the convolutional neural network and thus optimize neural networks with the guidance of probabilistic theory. Lastly, we will show many experiments to prove our model is really useful for computer vision in both simulation data and real data.

II. Related Work

A. Deep learning for motif inference

The deep learning method for motif inference can be categorized into two groups – CNN-based and RNN-based methods. As for the CNN-based model(may contain RNN), DeepBind [8] is the first CNN-based model to predict DNA-protein binding and since then deep learning has been widely utilized in this field for its great performance. [23] shows that deploying more convolutional kernels is always beneficial. iDeepA [24] applies an attention mechanism to automatically search for important positions. DeeperBind [25] and iDeepS [14] add an LSTM layer on DeepBind to learn long dependency within sequences to further improve the prediction performance. As for the RNN-based model, KEGRU [26] identifies TF binding sites by combining the Bidirectional Gated Recurrent Unit (GRU) network with k-mer embedding. Besides model selection, CONCISE [27] and iDeep [28] integrate other information(e.g. structured information) into predicting RBP-binding

[†]Equal Contribution. [*]Corresponding author.

978-1-6654-0898-1/21/$31.00 ©2021 IEEE
DOI 10.1109/ICTAI52525.2021.00032

sites and preference. Other work includes data augmentation [12], circular filters [29] and convolutional kernel networks [15], [16], [30].

B. Convolutional neural networks

Convolutional Neural Networks (CNNs), a special type of Neural Networks, have shown significant performance improvement in several Image Processing and Computer Vision competitions, such as ImageNet [6]. The powerful and effective learning ability of deep CNN is mainly derived from the utilization of multiple feature extraction stages that can automatically learn feature representations from the original images. In 2012, A. Krizhevsky et al. [6] drew attention to the public by AlexNet network which achieved a top-5 error rate of 15.3% outperforming the previous best traditional model in the ImageNet dataset. Since the 2012 milestone, many researchers have tried to go deeper into the sequences of convolution layers to achieve better performance. In 2014, K. Simonyan & A. Zisserman [31] introduced the 16-layers VGG model that chains multiple convolution layers to win this competition. The same year, M. Lin et al. [32] has developed the concept of "inception modules" which is further exploited by C. Szegedy et al. [33] who proposed a deeper network called GoogLeNet with 22 layers. The main common trend to design convolutional neural network models is the increasing network depth. However, as the depth increased, networks involved an increasing error rate due to the difficulties of optimizing extremely deep models. K. He et al. [34] proposed ResNet network by residual learning to further deepen the network to 101 layers. To avoid the tedious network architectures design, B. Zoph and Q.V. Le [35] proposed a new concept called Neural Architecture Search (NAS) which searches state-of-the-art networks automatically. After that, various NAS networks have been released to the community, such as PNAS [36], ENAS [37], EfficientNet [38] and so on. And these classic network architectures further become the backbone networks in other tasks, such as image retrieval [2] and object detection [39].

III. Materials and methods

A. Logo image

In the field of computer vision, a task similar to motif finding is to detect whether images have a specific logo. The task of detecting logo images is to predict whether an image contains a specific logo. The position of the logo called motif in each image is uncertain. In this way, we can promote predicting DNA-protein binding to a two-dimensional situation. For example, Figure 1 shows a positive and a negative sample respectively, given the example image motif is a cross.

Assume we have a black and white image, which is divided into $M \times N$ pixels. Similar to the motif finding problem, we represent it by a $M \times N$ matrix $\mathbf{X} = (x_{i,j})$ with one channel, and the value of a pixel is 0 or 1. $x_{i,j} = 1$ if the pixel in at the ith row and jth column is white;otherwise,$x_{i,j} = 0$.

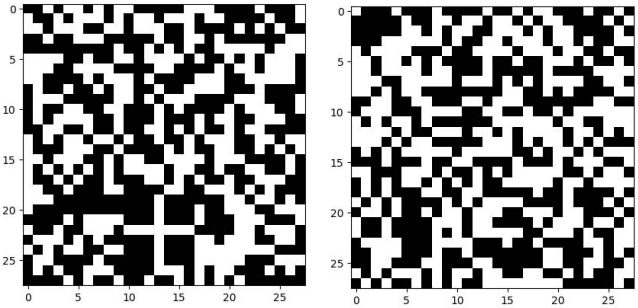

Fig. 1. Left: A positive sample. Right: A negative sample.

Every single channel image has a complementary matrix $\mathbf{X}'$ whose value of each pixel of an image is opposite to the value of the corresponding pixel of the original image (i.e. $\mathbf{X}'(i,j) = 1 - \mathbf{X}(i,j), \forall i,j$) to present the probability of the pixel to be black. For simplicity, we can only consider the first matrix and omit its complementary matrix. In this way, with the assumption that every pixel of an original image in the region of a motif is independent and obeys specific Bernoulli distribution, every pixel outside the region of motif obeys background distribution, we can calculate log-likelihood:

$$\begin{aligned} &\Pr(\text{Image}|\text{ Model }) \\ &= \prod_{(i,j)\notin \text{MR}} \Pr\left(x_{i,j}|\text{BG}\right) \prod_{(i,j)\in \text{MR}} \Pr\left(x_{i,j}|\text{MR}\right). \end{aligned} \tag{1}$$

where MR refers to the region of motif which is assumed to be square, BG refers to background distribution which is often assumed to be uniform distributed, and $x_{i,j}$ refers to value (i.e 0 or 1) of a pixel at position (i,j).

If we ignore background distribution, we can have a simple form of log-likelihood, which has related to the convolutional operation. Position weight matrix (PWM) can be defined to be a square matrix that records the probability of the value of each pixel equals 1 in the region of a motif. That is to say, convolutional operation obtains scores of candidate regions of a motif. With the idea above, a classification model of the convolutional neural network can be designed.

B. A parameterized convolutional neural network architecture

As shown in Figure 2, the model of neural network for this task is quite simple. The input is a $M \times N$ matrix where M and N denote the size of the image. Each pixel of the image is 0 or 1.

The first layer of the network is a 2-D convolutional layer, which can be considered as a motif scanner. The predetermined size of filters needs to correspond to the size of the region of the motif and usually needs to be larger. The output of each neuron on a convolutional layer is the convolution of the kernel matrix and the part of the input within the neuron's window size. Usually, the window size needs to be an odd number. An activation layer follows to filter negative scores which may disturb results.

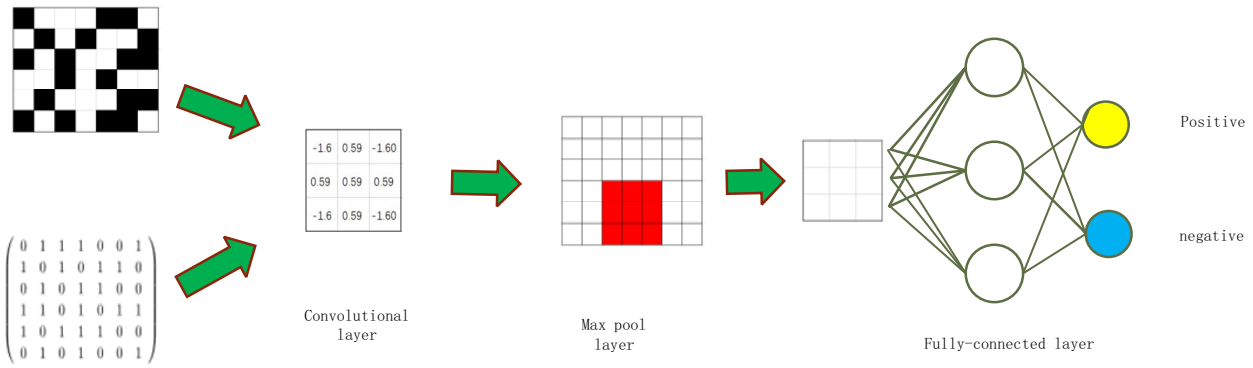

Fig. 2. Architecture of convolutional neural networks

The second layer is a global max-pooling layer, one for each convolutional layer(i.e each filter). Each of these max-pooling layers only outputs the maximum value of all of its convolutional layer outputs. This operation helps to identify whether the motif modeled by each convolutional operation exists in the input image or not.

The last layer is a fully-connected layer of size 1, with sigmoid activation, which corresponds to the last results, positive or negative. A probability of being a positive sample can arrive. Besides, another fully connected layer may be added to be the third layer if the model desired to need to be complex. Hidden layers are aimed to increase models' ability of learning complex tasks. And a dropout layer [40] can be chosen on the third layer to randomly mask portions of its output to avoid overfitting.

C. Probability interpretation of the model

For simplicity, we assume that all the positive samples share the same and only one motif. The number of filters is set to 1, and hidden layers are omitted. Our discussion can be promoted to more complicated situations similarly.

1) Transformation between kernel and PWM: In this part, we describe an exact kernel-to-PWM transformation, which consists of the following steps:

1) Assume we have a kernel $\mathbf{W}$.
2) Choose an arbitrary base of logarithm $b(b > 1)$. There is no other restriction on b.
3) Calculate the exponential transformed kernel $\mathbf{C}$, that is $c_{i,j} = b^w_{i,j}$, for all $\{i, j\}$ in $\{1, 2, \cdots, L\}$.
4) Normalize matrix $\mathbf{C}$ with its complementary matrix to get the PWM matrix $\mathbf{P}(\mathbf{W}, b)$=$\frac{c_{i,j}}{1+c_{i,j}}$.

Then by Theorem 1 in the next section, we have:

$$(\mathbf{X} * \mathbf{W})_{i,j} = \log_b \Pr(\mathbf{X}[i : (i + L - 1), j : (j + L - 1)]|\mathbf{P}(\mathbf{W}, b)) + d(\mathbf{W}, b)$$

In other words, the convolution by $\mathbf{W}$ is exactly the sum of a constant and the log-likelihood of the PWM transformed on $\mathbf{X}$.

On the other hand, from a PWM(with the restriction that no 0's or 1's are present), a kernel that is generated by the following steps is capable of classifying pictures in the same way as the PWM.

1) Assume we have a PWM $\mathbf{P}$.
2) Choose an arbitrary base of logarithm $b(b > 1)$. There is no other restriction on b.
3) Normalize matrix $\mathbf{P}$,we can simply taking $\mathbf{C} = \mathbf{P}$.
4) Calculate the logarithm transformed $\mathbf{W}$, that is $w_{i,j} = \log_b c_{i,j}$, for all i, j in $\{1, 2, \cdots, L\}$.
5) Add a positive shift to make sure all elements are non-negative if necessary.

If the positive simple obeys some certain PWM in the region of motif, one kernel can be calculated from PWM, which is called the exact kernel. Besides when testing, convolutional operation obtains scores of candidate motif regions which form a score-matrix. The activation layer is utilized a negative score which corresponds to deleting improper filters. Pooling operation retains the biggest score in some certain regions and the score corresponds to log-likelihood. Because of the bigger log-likelihood, the bigger score is, by some trained linear transformation which is monotonous, the score is transformed into log-ratio before sigmoid activation, and then into the probability of being a positive sample. From log-likelihood to the last probability, monotone transformation keeps that the bigger log-likelihood, the bigger last probability of being a positive sample, which is desired.

2) Hypothesis testing: Here, we use the view of the hypothesis testing to interpret our CNN. From the architecture of a neural network, we find that because our last classification is whether the probability is bigger than 0.5, the classification is equal to whether the biggest score is larger than some trained threshold. The null hypothesis can be set to be :

$$H_0 : \text{label of sample is positive.} \quad \text{vs}$$
$$H_1 : \text{label of sample is negative.}$$

The test statistic T is the biggest (i.e.score the output of pooling layer). The rejection region is $\{T < k\}$, where k is a trained threshold.

With the constraint of type I error, type II error can be compared between different models. What's more, the learned kernel corresponds to the test statistic T, and the weight of the fully connected layer corresponds to the trained threshold. Given a testing level α, we can learn a maximal k so that $\Pr(T < k) = \alpha$ when H_0 is true.

3) Bayesian view: From the Bayesian view, the learning weights of a neural network are similar to learning prior information without a test sample. A training set is like a model's teacher and gives tutors without dependence on testing samples. Combined with test samples, posterior distribution can arrive and we can get results.

4) Exact kernel: Exact kernel can be transformed from true PWM and in the contrast, PWM can also be obtained from the learned kernel. The 1-D situation has been proved for predicting DNA-protein binding [41], which can be promoted to a 2-D situation. If a true model (i.e. true PWM)exists it's evident that using log-likelihood combined with true PWM is the best testing method. As a result, kernel transformed from true PWM is the most effective kernel from the convolutional neural network. Without consideration of scale, the exact kernel has the form of

$$K(i,j) = \log P(i,j) - \log 1/2, \forall i,j$$

5) Upper bound of accuracy: Assume we know the region and exact value of motif. Then, we can give an estimation of the upper bound of the accuracy. We use a PWM matrix $\mathbf{P} = p_{i,j}$, $(i,j) \in \{1,2,\cdots,L\}$ to present a motif of size L. Moreover, we use the kernel described above, that is $K(i,j) = \log p_{i,j} - \log 1/2, \forall i,j$. Then our score of convolution is:

$$S = \sum_{i=1}^{L}\sum_{j=1}^{L} X_{i,j} \log 2p_{i,j}$$

where $X_{i,j}$ are independent random variables generated from Bernoulli distribution $B(1, p_{i,j})$. So the accuracy of classification is the probability of $\Pr(S \geq k)$, which is the probability of S bigger than k. Where k is learned from the hypothesis test described above. As all $p_{i,j}$ is known, the distribution of S is known and we can approximate it by numerical methods and estimate it.

6) Benefits of multiple kernels for convolutional neural network: Kernels can be called detectors and convolutional operation play the same role as "motif scan" operation in a PWM-based model. However, as the discussion above, the coefficients of motif detector matrices are arbitrary and under no hard constraints but correspond to the coefficients of specific PWM.

Deep neural networks perform best when trained with more parameters (in our case, motif detectors) than is exactly required to model the data, which is because of the way models are randomly initialized at training start, and to the uncertainty of gradient descent learning. Besides, models of neural networks are not convex, which can't promise loss function can converge to a globally optimal solution rather than a locally optimal solution. For example, if an image has one true motif, training with one kernel will often fail to model the data and get accurate results, whereas training with 16 kernels will succeed much more frequently to get accurate results. However, in this case, we may not directly get true PWM because the information of true PWM may be divided into several motif detectors. For the 1-D situation, it has been shown that more motif detectors lead to better and robust results [23].

7) Size of motif detectors: The size of motif detectors is a hyper-parameter that needs to be pre-decided. The size is related to the true size of the motif. However, if the size pre-decided is small than the real size, it may lead to failing to model the data when the number of motif detectors is not enough, while smaller motifs can still be learned with irrelevant positions having near-zero coefficients. As a result, often we choose the size of kernels is 2 times larger than the estimated size. Too large kernels may lead to much more calculation for training data, which is consuming.

8) Calculating PWMs: There are two methods to get PWMs. One method is utilizing exact transform for the kernel to get PWMs. Another method is the same as DeepBind [8]. We can generate a PWM from a detector's(i.e. kernel) response to actual sequences. We feed all sequences from the testing set through the convolutional layer and align all the sequences that passed the given threshold. And then position frequency matrix can be obtained and approximate PWM.

9) Multiple convolutional layers: In the field of image recognition, usually deeper networks perform better than networks with wider filters [42], which is the reason that size of filters usually is set to be 3×3 or 5×5. Deeper networks have wider receptive field [43] while a simple model outperforms in motif discovery [23]. Multiple convolutional layers without activation are equal to only one convolutional layer with the size of filters same as the size of the receptive field. The only difference is that multiple convolutional layers add additional unequal weights automatically to filters. Generally, with the ReLU activation layer, multiple convolutional layers can learn more integrated information from complex models while the true rules of motif detecting only need one convolutional layer. Adding more convolutional layers increases the complexity of neural networks, deviates the embedded true rules of the model, and sacrifices the good interpretability of the model.

D. Gibbs sampling for general images

For general images whose values of pixels are between 0 and 1, Gibbs sampling [44] can be utilized. For the value of each pixel, we can consider the value to be the probability of the value being 1 with the assumption that values of pixels obey the Bernoulli distribution. Then for every pixel, we can sample logo images with values 0 or 1 according to the probability and thus generate a large number of images. For example, if a value of one pixel is 0.8, we can sample this pixel with the assumption that value obeys Bernoulli distribution with probability 0.8, and if we sample a lot of times, the expectation of frequency at this pixel being 1 is 0.8 according to the law of large numbers [45]. If a real-world image contains

many logos, we can find that the expectation of scores equals the neural network output with the input. In this way, our model can also be promoted to deal with general images.

IV. Theoretical Proof

Let $\mathbf{A} = (a_{i,j})$ be a matrix. We use $\mathbf{A}[i_1 : i_2, j_1 : j_2]$ to represent the sub-matrix of $\mathbf{A}$ generated by extracting rows from i_1 to i_2, and columns from j_1 to j_2. We use a matrix $\mathbf{W}$ to represent an arbitrary kernel. The shape of $\mathbf{W}$ is usually $L \times L$, where L is the size of the kernel.

We use a matrix $\mathbf{X}$ to represent an arbitrary input black and white picture. The picture is divide into $M \times N$ grids. In our model, we simply assume $\min\{M, N\} \geq L$, that is the input pictures are not smaller than the kernels.

As for every grid of the picture is either black or white, the matrix $\mathbf{X}$ can be written as:

$$x_{i,j} = \begin{cases} 1, & \text{if the grid at the } i\text{th row and } j\text{th column is white} \\ 0, & \text{otherwise.} \end{cases}$$

We use a $L \times L$ matrix $\mathbf{P}$ called PMW to represent any motif of size L in a picture. It should be a PWT(Position Weight Tensor) with two layers. The second layer of the tensor is complementary to the first one because the picture is black and white. Thus we can simplify tensor to a matrix for convenience. For the (i, j)-th element of $\mathbf{P}$, $p_{i,j} = \Pr(\text{the } i\text{-th row and } j\text{-th column of the motif are white})$.

We define $\mathbf{X} * \mathbf{W}$ as the discrete convolution with matrix $\mathbf{W}$ on matrix $\mathbf{X}$.

$$(\mathbf{X} * \mathbf{W})_{i,j} := \sum_{u=1}^{L} \sum_{v=1}^{L} x_{i+u-1,j+v-1} w_{u,v}$$

We define $\Pr(\mathbf{X}|\mathbf{P}(\mathbf{W}, b))$ as the probability that the picture represent by matrix $\mathbf{X}$ is generated from the PWM $\mathbf{P}(\mathbf{W}, b)$ with Bernoulli distribution.

Theorem1. Given $\mathbf{W}, b > 0$, we can transform them into the PWM $\mathbf{P} = \mathbf{P}(\mathbf{W}, b)$ use the method described above. Then, $\forall \mathbf{X}$, we have:

$$(\mathbf{X} * \mathbf{W})_{i,j} = \log_b \Pr(\mathbf{X}[i : (i + L - 1), j : (j + L - 1)] \mid \mathbf{P}(\mathbf{W}, b)) + d(\mathbf{W}, b)$$

where $d(\mathbf{W}, b)$ is a constant only depends on $\mathbf{W}$ and b.

Proof. By the definition of PWM and the independence, a picture is generate from a PWM is the production of probability of "the color at each position in that picture" at the same position in the PWM. We use the following formula:

$$\begin{aligned} &\Pr(\mathbf{X}[i : (i + L - 1), j : (j + L - 1)] \mid \mathbf{P}(\mathbf{W}, b)) \\ &= \prod_{\{u,v\} \in T_1} (\mathbf{P}(\mathbf{W}, b)_{u,v}) \prod_{\{u,v\} \in T_2} (1 - \mathbf{P}(\mathbf{W}, b)_{u,v}) \end{aligned}$$

where

$$\begin{aligned} T_1 &= \{(u, v) | (u, v) \in [1, 2, \cdots, L] \times [1, 2, \cdots, L]; x_{i+u,j+v} = 1\}, \\ T_2 &= \{(u, v) | (u, v) \in [1, 2, \cdots, L] \times [1, 2, \cdots, L]; x_{i+u,j+v} = 0\}. \end{aligned}$$

Taking the logarithm with base b, we can get:

$$\begin{aligned} &\log_b \Pr(\mathbf{X}[i : (i + L - 1), j : (j + L - 1)] \mid \mathbf{P}(\mathbf{W}, b)) \\ &= \sum_{\{u,v\} \in T_1} \log_b (\mathbf{P}(\mathbf{W}, b)_{u,v}) + \sum_{\{u,v\} \in T_2} \log_b (1 - \mathbf{P}(\mathbf{W}, b)_{u,v}) \end{aligned} \tag{2}$$

On the other hand, from the convolution we have:

$$(\mathbf{X} * \mathbf{W})_{i,j} = \sum_{u=1}^{L} \sum_{v=1}^{L} x_{i+u-1,j+v-1} w_{u,v} = \sum_{\{u,v\} \in T_1} w_{u,v}.$$

By the transformation way descried above, we have $w_{u,v} = \log_b c_{u,v}$ and $\mathbf{P}(\mathbf{W}, b) = \frac{c_{i,j}}{1 + c_{i,j}}$.

$$\begin{aligned} (\mathbf{X} * \mathbf{W})_{i,j} &= \sum_{\{u,v\} \in T_1} \log_b \frac{\mathbf{P}(\mathbf{W}, b)_{u,v}}{1 - \mathbf{P}(\mathbf{W}, b)_{u,v}} \\ &= \sum_{\{u,v\} \in T_1} \log_b \mathbf{P}(\mathbf{W}, b)_{u,v} \\ &\quad - \sum_{\{u,v\} \in T_1} \log_b (1 - \mathbf{P}(\mathbf{W}, b)_{u,v}). \end{aligned}$$

As the definition of sets T_1 and T_2,we know that $T_1 \cap T_2 = \emptyset$ and $T_1 \cup T_2 = [1, 2, \cdots, L] \times [1, 2, \cdots, L]$. That is:

$$\begin{aligned} (\mathbf{X} * \mathbf{W})_{i,j} &= \sum_{\{u,v\} \in T_1} \log_b (\mathbf{P}(\mathbf{W}, b)_{u,v}) \\ &\quad + \sum_{\{u,v\} \in T_2} \log_b (1 - \mathbf{P}(\mathbf{W}, b)_{u,v}) \\ &\quad - \sum_{i'=1}^{L} \sum_{j'=1}^{L} \log_b (1 - \mathbf{P}(\mathbf{W}, b)_{i',j'}). \end{aligned} \tag{3}$$

Comparing Eq.2 and Eq.3, we can finish the proof by setting $d(\mathbf{W}, b) = -\sum_{i'=1}^{L} \sum_{j'=1}^{L} \log_b (1 - \mathbf{P}(\mathbf{W}, b)_{i',j'})$.

V. Experiments and discussion

A. Simulation data

In this section, we verify our model by simulation data. True PWM is below, whose shape is like a cross. Background distribution is assumed to be Bernoulli distribution with a probability 0.5. Label 1 represents that the image has the given motif, while label 0 represents that pixels of the image are exactly random. The proportion of positive samples is half in both the training set and in the testing set. The size of the convolutional filter is 7×7, which is exactly equal to the size of the motif. Our given square PWM is shaped like a cross with probability p and other grids of a motif with probability $1 - p$. Three experiments are conducted with $p = 0.99, 0.9$ and 0.8 respectively.

First, we choose the simplest neural networks with only one kernel in the convolutional layer and without a hidden layer. The exact kernel can be obtained from PWM. First, a great accuracy can be obtained if an exact kernel is utilized because only two parameters of the last layer need learning. In this situation, we believe the upper bound of accuracy can

TABLE I
ACCURACY OF MODELS IN SIMULATION DATA

Different probability in PWM	Exact kernel	Learned kernel
0.99/0.01	1.0000	1.0000
0.9/0.1	0.9360	0.9690
0.8/0.2	0.7040	0.7760

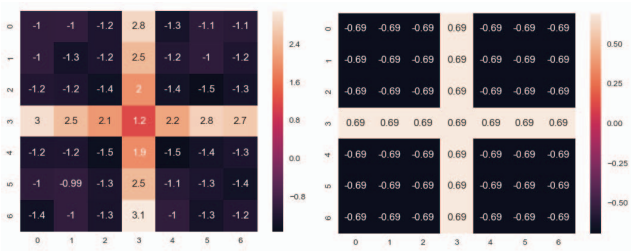

Fig. 3. Trained kernel (Left) and exact kernel (Right) with $p = 0.8$

be obtained because parameters can be fully learned without consideration of overfitting or under-fitting. And then we utilize one variable kernel and train a whole model.

We found that by trained kernel will approximate the exact kernel without scaling. At least, the positive and negative numbers in both kernels are corresponding. Sometimes learned kernel may have a gap to the exact kernel, which is considered as the best kernel. This fact shows that training images with one motif with only one kernel may lead to poor results which are not desired. Because simulation data has a true model, our accuracy obtained with the exact kernel is considered to approximate the upper bound of this model with the Monte Carlo method. What's more, the exact kernel results in a faster convergence, which is expected for fewer parameters.

However, we found that the learned kernel will result in more accuracy than the exact kernel, which is out of our expectation. We guess that because of the pseudo-randomness of simulation data, deep learning studies this property which results in high accuracy. For a lower accuracy with $p = 0.8$, we will utilize this data to compare different models of neural networks. what's more, the learned kernel has a symmetrical structure, and for the cross, the value of the center is somewhat smaller than the value of boundary, which we guess results from the influence of points around.

B. Comparison for different numbers of filters

Next, we test with our simulation data for comparison of different numbers of filters. For one model, only one kernel is utilized and for another, 32 kernels are utilized in a convolutional layer. The accuracy for the two models is 0.7760 and 0.7980 respectively. It's shown that although multiple kernels can't have a direct transformation to real PWM, they have benefits to increasing accuracy, which verifies what we discuss above. What's more, we advise that to reduce the amount of calculation, we prefer to increasing the number of filters rather than the size of filters because the latter may cause a large calculation.

C. Comparison with or without multiple convolutional layers

For multiple convolutional layers, to keep the same size of the receptive field, we utilize 3 convolutional layers with filters size 3. For each layer, the number of filters is 16. The number of parameters for the whole model is 4769 while the one-layer model has 801 parameters. However, the accuracy for models of multiple convolutional layers is 0.7680 lower than the original accuracy 0.7980, which shows that more parameters lead to a bad result.

This result is similar to the 1-D result for DNA-binding prediction [23]. For the general task of image recognition, for the same size of the receptive field, a deeper model of the neural network may have better accuracy. We believe for this task with a true model, one convolutional layer has statistical interpretation which multiple convolutional layers with activation ruin this architecture. What's more, multiple convolutional layers bring in a large calculation which disrupts us.

D. Results of real data

In this part, we will utilize our model in a real data set. The MNIST digit classification task is composed of 28×28 images of the 10 handwritten digits [46].

Our model has only one convolutional layer with 128 filters of size 15×15 and a fully connected layer of size 256. A dropout layer [40] with a rate 0.9 is used on the third layer output to avoid overfitting. The last layer is also a fully connected layer with a size 10. The soft-max activation will arrive at the probability of different labels. We believe that every label of digits has its motifs, which is reasonable. For example, digit eight has two circles which can be considered as its motifs. We train our model with 30000 epochs, and 0.9563 accuracy can be obtained, which shows that our neural network can deal with the specific task of image recognition. What's more, with a probabilistic interpretation of our model, our model of the neural network is good at dealing with images with noise, which shows the good robustness of our model. The MNIST digit classification task is not typical enough to show our model's advantages.

VI. CONCLUSION

In this paper, we successfully promote the motif finding problem of sequences to a 2-D situation and raise a new task for image classification – motif finding problem of logo images. The model of DeepBind is still utilized and promoted to a 2-D situation. Because of the good interpretation of DeepBind, we can get convolutional neural networks an interpretation in this model. Statistical ideas combined with deep learning give guidance to the adjustment of hyper-parameters and structure design of neural networks. Since deep learning can get accurate results in many classification problems, recently it has much successful application in the field of computational biology [47]–[49]. Statistical Learning can be combined with deep learning for specific models, which may help open up the black box of deep learning. We believe our interpretation for

the convolutional neural network can be promoted and applied to help improve the results obtained.

ACKNOWLEDGMENT

This work was supported by The National Key Research and Development Program of China (No.2016YFA0502303) and the National Natural Science Foundation of China (No.31871342).

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.
[2] X. Luo, D. Wu, Z. Ma, C. Chen, J. Ma, M. Deng, Z. Jin, J. Huang, and X.-s. Hua, "Cimon: Towards high-quality hash codes," in *IJCAI*, 2021.
[3] X. Luo, C. Chen, H. Zhong, H. Zhang, M. Deng, J. Huang, and X. Hua, "A survey on deep hashing methods," *arXiv preprint arXiv:2003.03369*, 2020.
[4] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
[5] S. Hochreiter and J. Schmidhuber, "Lstm can solve hard long time lag problems," in *Advances in neural information processing systems*, 1997, pp. 473–479.
[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[8] 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 biotechnology*, vol. 33, no. 8, p. 831, 2015.
[9] J. Zhou and O. G. Troyanskaya, "Predicting effects of noncoding variants with deep learning–based sequence model," *Nature methods*, vol. 12, no. 10, p. 931, 2015.
[10] P. C. B. Staff, "Correction: Enhanced regulatory sequence prediction using gapped k-mer features," *PLoS computational biology*, vol. 10, no. 12, p. e1004035, 2014.
[11] T. N. Sainath, A. R. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for lvcsr," in *IEEE International Conference on Acoustics*, 2013.
[12] Z. Cao and S. Zhang, "Simple tricks of convolutional neural network architectures improve dna–protein binding prediction," *Bioinformatics*, 2018.
[13] X. Pan and H.-B. Shen, "Predicting rna-protein binding sites and motifs through combining local and global deep convolutional neural networks," *Bioinformatics*, 2018.
[14] X. Pan, P. Rijnbeek, J. Yan, and H.-B. Shen, "Prediction of rna-protein sequence and structure binding preferences using deep convolutional and recurrent neural networks," *BMC genomics*, vol. 19, no. 1, p. 511, 2018.
[15] X. Luo, W. Chi, and M. Deng, "Deepprune: Learning efficient and interpretable convolutional networks through weight pruning for predicting dna-protein binding," *Frontiers in genetics*, vol. 10, p. 1145, 2019.
[16] X. Luo, X. Tu, Y. Ding, G. Gao, and M. Deng, "Expectation pooling: an effective and interpretable pooling method for predicting dna–protein binding," *Bioinformatics*, vol. 36, no. 5, pp. 1405–1412, 2020.
[17] Y. Guo, X. Luo, L. Chen, and M. Deng, "Dna-gcn: Graph convolutional networks for predicting dna-protein binding," in *ICIC*, 2021.
[18] D. Castelvecchi, "Can we open the black box of ai?" *Nature News*, vol. 538, no. 7623, p. 20, 2016.
[19] T. Ching, D. S. Himmelstein, B. K. Beaulieu-Jones, A. A. Kalinin, B. T. Do, G. P. Way, E. Ferrero, P.-M. Agapow, M. Zietz, M. M. Hoffman *et al.*, "Opportunities and obstacles for deep learning in biology and medicine," *Journal of The Royal Society Interface*, vol. 15, no. 141, p. 20170387, 2018.
[20] J. Choo and S. Liu, "Visual analytics for explainable deep learning," *arXiv preprint arXiv:1804.02527*, 2018.
[21] R. Shwartz-Ziv and N. Tishby, "Opening the black box of deep neural networks via information," *arXiv preprint arXiv:1703.00810*, 2017.
[22] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," 1999.
[23] H. Zeng, M. D. Edwards, G. Liu, and D. K. Gifford, "Convolutional neural network architectures for predicting dna–protein binding," *Bioinformatics*, vol. 32, no. 12, pp. i121–i127, 2016.
[24] X. Pan and J. Yan, "Attention based convolutional neural network for predicting rna-protein binding sites," *arXiv preprint arXiv:1712.02270*, 2017.
[25] H. R. Hassanzadeh and M. D. Wang, "Deeperbind: Enhancing prediction of sequence specificities of dna binding proteins," in *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2016, pp. 178–183.
[26] Z. Shen, W. Bao, and D.-S. Huang, "Recurrent neural network for predicting transcription factor binding sites," *Scientific reports*, vol. 8, no. 1, p. 15270, 2018.
[27] Ž. Avsec, M. Barekatain, J. Cheng, and J. Gagneur, "Modeling positional effects of regulatory sequences with spline transformations increases prediction accuracy of deep neural networks," *Bioinformatics*, vol. 34, no. 8, pp. 1261–1269, 2017.
[28] X. Pan and H.-B. Shen, "Rna-protein binding motifs mining with a new hybrid deep learning based cross-domain knowledge integration approach," *BMC bioinformatics*, vol. 18, no. 1, p. 136, 2017.
[29] C. F. Blum and M. Kollmann, "Neural networks with circular filters enable data efficient inference of sequence motifs," *Bioinformatics*, 2019.
[30] D. Chen, L. Jacob, and J. Mairal, "Biological sequence modeling with convolutional kernel networks." *Bioinformatics (Oxford, England)*, 2019.
[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[32] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.
[33] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[35] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016.
[36] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 19–34.
[37] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," *arXiv preprint arXiv:1802.03268*, 2018.
[38] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.
[39] J. Han, D. Zhang, G. Cheng, N. Liu, and D. Xu, "Advanced deep-learning techniques for salient and category-specific object detection: a survey," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 84–100, 2018.
[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
[41] Y. Ding, J.-Y. Li, M. Wang, and G. Gao, "An exact transformation of convolutional kernels enables accurate identification of sequence motifs," *bioRxiv*, p. 163220, 2018.
[42] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.
[43] D. T. Jones and S. M. Kandathil, "High precision in protein contact prediction using fully convolutional neural networks and minimal sequence features," *Bioinformatics*, 2018.
[44] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *Readings in Computer Vision*, vol. 20, no. 5-6, pp. 25–62, 1987.
[45] M. Loève, *Independent Identically Distributed Summands*, 1977.
[46] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[47] C. Angermueller, T. Pärnamaa, L. Parts, and O. Stegle, "Deep learning for computational biology," *Molecular systems biology*, vol. 12, no. 7, p. 878, 2016.

[48] L. Rampasek and A. Goldenberg, "Tensorflow: Biology's gateway to deep learning?" *Cell systems*, vol. 2, no. 1, pp. 12–14, 2016.
[49] S. Min, B. Lee, and S. Yoon, "Deep learning in bioinformatics," *Briefings in bioinformatics*, vol. 18, no. 5, pp. 851–869, 2017.