IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 21, 2020, accepted May 12, 2020, date of publication May 19, 2020, date of current version June 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2995762

BP-GAN: Interpretable Human Branchpoint Prediction Using Attentive Generative Adversarial Networks

HYEONSEOK LEE[1], SANGWOO YEOM[1], AND SUNGCHAN KIM[1,2]
[1]Division of Computer Science and Engineering, Jeonbuk National University, Jeonju 54896, South Korea
[2]Research Center for Artificial Intelligence Technology, Jeonbuk National University, Jeonju 54896, South Korea

Corresponding author: Sungchan Kim (s.kim@jbnu.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (MSIT) under Grant 2019R1F1A1061941.

ABSTRACT Branchpoints (BPs) are essential sequence elements of ribonucleic acids (RNAs) in splicing, which is the process of creating a messenger RNA (mRNA) that is translated into proteins. This study proposes to develop deep neural networks for BP prediction. Extensive previous studies have shown that the existence of BP sites depends on sequence patterns called motifs; hence, the prediction model must accurately explain its decisions in terms of motifs. Existing approaches utilized either handcrafted features for interpretable, though less accurate, predictions or deep neural networks that were accurate but difficult to explain. To address the aforementioned difficulties, the proposed method incorporates 1) generative adversarial networks (GANs) to learn the latent structure of RNA sequences, and 2) an attention mechanism to learn sequence-positional long-term dependency for accurate prediction and interpretation. Our method achieves highly satisfying results in various performance metrics with adequate interpretability. We demonstrated that, without any prior biological knowledge, BP prediction by the proposed method is closely related to three motifs, the consensus sequence surrounding BPs, polypyrimidine tract, and 3' splice site, that are well-established in molecular biology.

INDEX TERMS Branchpoint prediction, deep neural networks, generative adversarial networks, interpretability.

I. INTRODUCTION

The human body has numerous types of cells, such as blood cells, neurons, and liver cells. Even though their functions are different, the cells are created from the same set of deoxyribonucleic acid (DNA) by combining different genes to synthesize a functional gene product, e.g., protein. [1] The process of making pre-messenger ribonucleic acid (pre-mRNA) from DNA, messenger RNA (mRNA) from pre-mRNA, and protein from mRNA is referred to as gene expression. [2]

Large intervening sequences called *introns* are *spliced out* and only the flanking sequences called *exons* are *spliced in* together to comprise the mRNA, as shown in Figure 1(a). This is the process of alternative splicing to generate various types of proteins from the same mRNA. As an essential part of gene expression, splicing incorporates three key nucleotides: 5' and 3' splice sites (sss) and a BP site. The 5' and 3' sss denote the first (upstream) and last (downstream) nucleotides within an intron, respectively. A BP is a nucleotide where the 5' ss is joined to a lariat-forming to make an intron separated from a pre-mRNA. After splicing, each mRNA molecule encodes the instructions to build proteins. Thus, identifying the 5' ss, 3' ss, and BP is crucial for understanding the mechanism of splicing. Mis-spliced mRNA can result in altered proteins and often damage their functions, such as causing diseases [2]. The 5' and 3' sss are detected by well-established RNA sequencing techniques [3]; however, it is difficult to identify BPs in sequencing experiments,

The associate editor coordinating the review of this manuscript and approving it for publication was Zijian Zhang.

[1]DNA is a long string of paired chemical molecules called *nucleotides* that are of four types denoted by A, C, G, and T [1]. Genetic information in DNA is organized into nucleotide sequences called genes. The set of genetic material stored in DNA is called a genome.

[2]RNA is a molecular transcript of DNA that conveys the genetic information to create proteins. The types of nucleotides in RNA are A, C, G, and U instead of T.

as the lariats are degraded rapidly and appear rarely. Recent efforts have enriched the lariats in RNA sequencing, building large-scale datasets of genome-wide BP annotations [4], [5]; however, only 40% of the entire human introns were covered by annotation of almost 130,000 human BPs.

In view of these difficulties, we propose a deep neural network model to predict the BP for a given RNA sequence taking into consideration the following: First, BP sites are known to co-exist with motifs or sequence patterns of up to tens of nucleotides that are typically not readable by humans and difficult to identify; a few ultra-conservative sequence motifs were observed experimentally [6]. Therefore, it is necessary to develop a model that explains its predictions in terms of motifs with high accuracy. Second, the distribution of BPs is highly biased within 20 to 50 nucleotides upstream from the 3' sss, as shown in Figure 1(a) [7], causing a class-imbalance in the datasets. The existing approaches were constrained as they relied on either interpretable but inaccurate handcrafted sequential features or deep neural networks of poor interpretability [3], [8]–[12].

We address these issues by using generative adversarial networks (GANs) [13], [14] to learn the latent structure of RNA sequences for BP prediction, which we call *BP-GAN* hereafter. The BP-GAN decomposes the underlying structure of input RNA sequences into two types of features: one directly affecting the BP prediction and the other irrelevant to the prediction. We further improve the BP-GAN by adopting a novel integration of attention mechanism and triplet loss with hard negative mining. The attention mechanism enables to learn long-term dependencies for a given input sequence for compelling sequence modeling. In addition, it provides a principled way of interpreting the regional clues regarding the part of a sequence that the BP-GAN attended to. As shown, the BP-GAN recovers three biologically meaningful motifs. To resolve the problem of class imbalance, we use triplet loss to regularize the BP-GAN during training. This improves the prediction accuracy effectively. In summary, the contributions of this study are as follows:

1) We present the BP-GAN as a novel integration of attention into GAN, regularized by triplet loss. Deep neural networks in the BP-GAN are carefully designed for BP prediction, achieving excellent results in two large-scale datasets.
2) Our method uses a GAN to learn the structure of latent variables for RNA sequences relevant to BP prediction. To the best of our knowledge, a GAN is suitable for generative tasks [15]–[19] in gene expression, however, this is the first GAN-based attempt to address a discriminative task. We believe that our approach is also applicable to other related downstream tasks.
3) As a key benefit, the BP-GAN provides a means to interpret and visualize the rationale behind its decisions by using attention, which was not addressed in previous studies on BP prediction. In particular, we demonstrated that it can recover three biologically meaningful motifs that are strong indicators of BPs.

II. RELATED WORK

First, we present an overview of the deep neural network-based approaches related to RNA splicing. Next, previous efforts on BP prediction are summarized to highlight our contributions.

A. GENERATIVE MODELS

A generative model aims to determine an underlying structure of variables relevant to real use-cases. For example, only small structured subsets of pixel values are plausible exemplar images in the real world. A GAN is a deep generative model with an exceptional capability to learn the structure of parameters and generate samples similar to true data [13]. It has initiated several recent studies in genomics [15]–[19]. In [15], [19], the approaches generated DNA sequences using a GAN. A GAN model in [17] aimed to generate RNA sequences for protein function analysis whereas other works focused on generating protein structures [16] and sequences [18]. Gene expression was investigated to simulate an RNA-seq dataset focusing on the diversity of skin cells [20]. This work interpreted the learned parameters in a biologically meaningful manner that was similar to ours. An approach in [21] used a GAN to learn the probabilistic distribution of gene expression in single cells.

B. ATTENTION MECHANISM

Attention learns inter-/intra-sequence dependencies in sequential modeling [22]–[24]. Encouraged by its success, a number of recent studies in genomics incorporated attention into various prediction models on RNA-protein binding sites [25], gene expression analysis [1], and precursor microRNAs [26]. These approaches primarily focused on attention at the level of hundreds of nucleotides [1], or used recurrent neural networks (RNNs) [26] to learn long-term dependencies between sequence elements [24]. However, we considered an attention mechanism without using RNNs, called *self-attention* or *intra-attention*, to relate the elements of an input sequence with each other more effectively.

C. MACHINE LEARNING APPROACHES FOR BP PREDICTION

Earlier studies on BP prediction were largely based on classical machine learning with handcrafted sequential features [8]–[11], or deep neural networks [3], [12], [27], [28]. The former included features such as sequence conservation and positional bias for a support vector machine [8], motifs of an intron sequence for an ensemble of multiple algorithms [9], and a score function calculated from position-specific scoring matrix (PSSM) and binding energy of spliceosome [10]. An approach similar to [10] solved the mixture model of motif interference and polypyrimidine tract[3] A gradient boosting algorithm was used in [12].

[3] A region of mRNA that promotes RNA splicing, which is 15-20 nucleotides long and rich with *C* and *U* nucleotides [29].

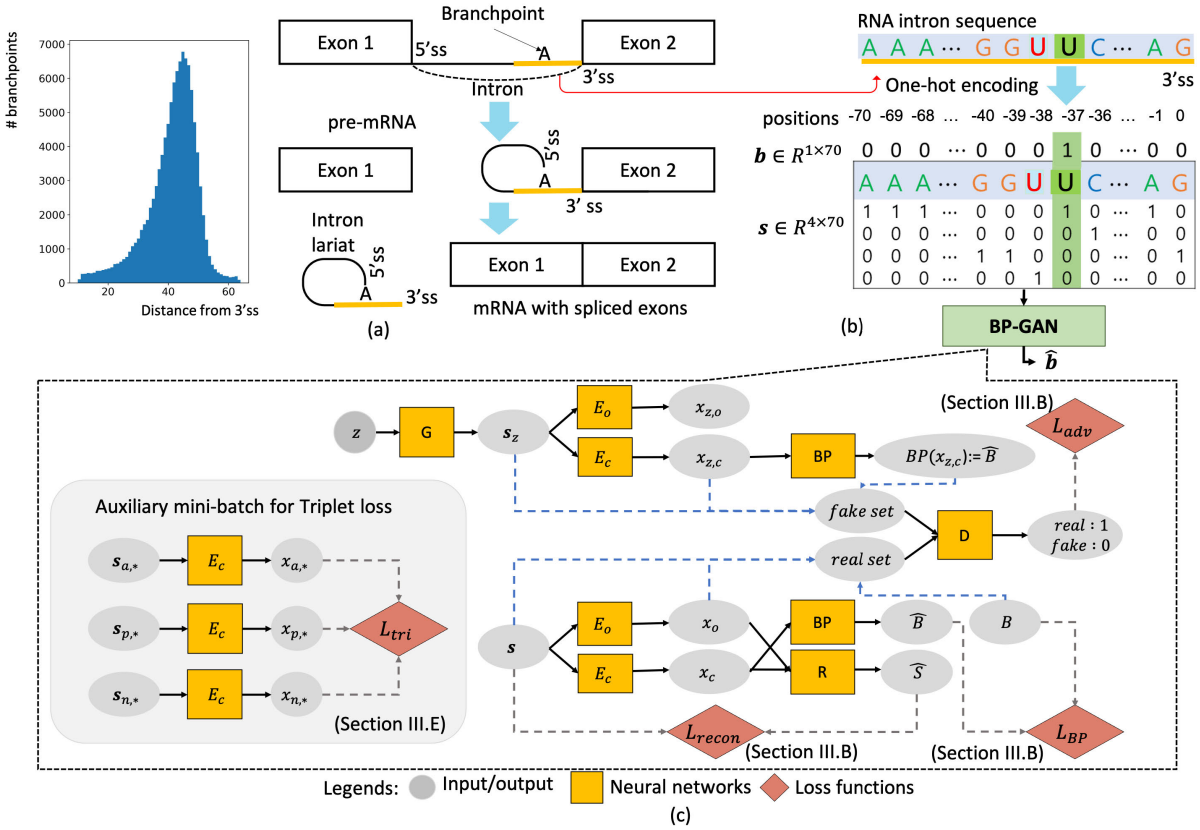

FIGURE 1. Overview of the proposed BP-GAN method: (a) (left) BP distribution of a dataset [5] and (right) the role of BPs in RNA splicing, (b) input format of RNA sequences, and (c) internal structure of the BP-GAN. The details of loss functions are given in Section III. A full description of network parameters and the auxiliary mini-batch using triplet loss is detailed in the Appendix.

On the other hand, [27] pioneered the use of convolutional neural networks (CNNs) with extra annotations of positional information of nucleotides. An approach proposed in [30] was similar to ours in that they visualized motifs for predicting splice sites in DNA sequences; however, it was based on a simple CNN that lagged behind state-of-the-art approaches in terms of prediction accuracy [27]. Recently, an RNN-based approach using bi-directional long short-term memory (LSTM) learned sequential features [3]. This work was extended in [28] by processing input sequences using CNN before applying them to LSTM and achieved excellent results. The RNN-based approaches used additional information on the binding energy of nucleotides whereas our approach requires only RNA sequences. To summarize, the existing deep neural network-based approaches are merely straightforward applications of CNN or RNN to BP prediction and few of the previous studies addressed the issue of interpretability of the predictions. The work in [31] presented the comprehensive evaluations on the recent approaches for various performance measures on in-house RNA datasets as well as public ones.

III. PROPOSED METHOD

A. PROBLEM FORMULATION

The BP-GAN takes as its input an intronic region in an RNA sequence of N nucleotides of the four types $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{U}\}^N$, as shown in Figure 1(b). We use a one-hot vector $s_i = \{0, \ldots, 0, 1, 0, \ldots, 0\} \in \mathbb{R}^4$ to represent each type of nucleotide and denote the input sequence by $\boldsymbol{s} = \{s_i\} \in \mathbb{R}^{4\times N}$. Following this notation, $s_N \in \boldsymbol{s}$ is the last nucleotide at the 3' ss. For training the BP-GAN, a ground-truth of BP sites for a given input $\boldsymbol{s}$ is given by $\boldsymbol{b} = \{b_i\} \in \{0, 1\}^N$. A sequence can have multiple BPs [5]. The BP-GAN predicts a vector $\hat{\boldsymbol{b}} = \{\hat{b}_i\} \in \{0, 1\}^N$ to represent the probability to be a BP for each nucleotide in $\boldsymbol{s}$.

B. LEARNING FEATURE REPRESENTATION USING GAN

A GAN learns generative models by utilizing the discriminative capability of neural networks. A generator G takes a latent variable z sampled from known prior P_z to generate a sample $\boldsymbol{s}_z = G(z)$. The learning process of a GAN is a minimax game of G parameterized as θ_G and a discriminator D with θ_D, which is formulated as

$$\begin{aligned}\min_{\theta_D}\max_{\theta_G} J(G, D) \\ &= L_{adv} \\ &= -\left\{\mathbb{E}_{\boldsymbol{s}\sim P_s}\left[logD(\boldsymbol{s})\right] + \mathbb{E}_{z\sim P_z}\left[log(1 - D(\boldsymbol{s}_z))\right]\right\}. \quad (1)\end{aligned}$$

The discriminator D tries to decrease the cost function $J(G, D)$ by maximizing the probability of a real data $\boldsymbol{s}$, which is sampled from the real distribution P_s, to be classified as real, $D(\boldsymbol{s})$, and classifying a fake data $\boldsymbol{s}_z = G(z)$ as fake

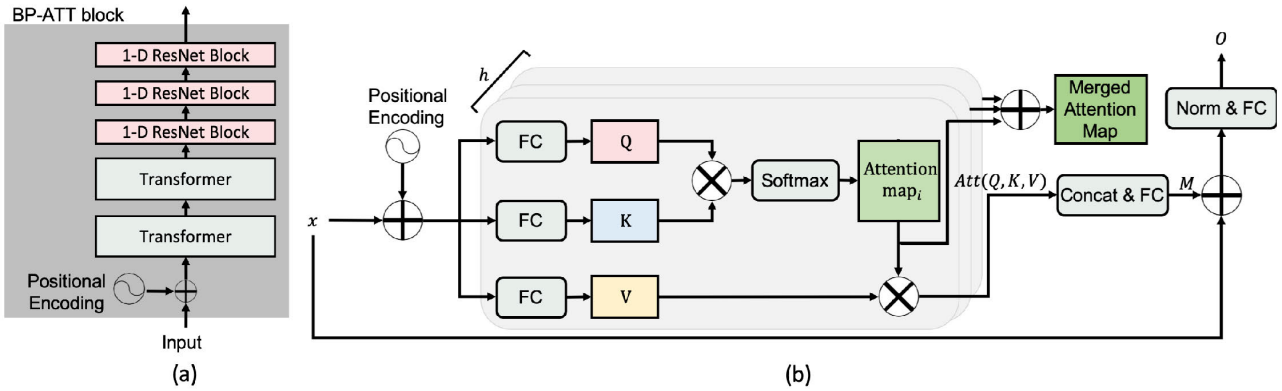

FIGURE 2. (a) Branchpoint-attention (BP-ATT) block and (b) the first Transformer of the BP-ATT block. A Transformer block is mainly decomposed into h identical attention layers. Each of the layers evaluates a scaled-dot product attention.

in a term denoted by adversarial loss L_{adv}. On the contrary, the generator G tries to mislead D, i.e., decrease $1 - D(\boldsymbol{s}_z)$, and thus, increase the cost function.

Figure 1(c) illustrates the structure of the BP-GAN. An input $\boldsymbol{s}$ is encoded into two latent variables $x_c = E_c(\boldsymbol{s})$ and $x_o = E_o(\boldsymbol{s})$ through two encoders E_c and E_o parameterized by θ_{E_c} and θ_{E_o} with $x_c, x_o \in \mathbb{R}^{d_x \times N}$, where d_x is a dimension of a latent variable corresponding to a sequence element. x_c represents a context feature that is similar in all sequences with the same BP sites. We consider two BP sites to be identical based on their positions, regardless of the type of nucleotide. x_o describes other aspects of the sequence irrelevant to the prediction of BPs. Thus, it is possible to reconstruct $\boldsymbol{s}$ from x_c and x_o. To this end, we use a network R with parameters θ_R: $\hat{\boldsymbol{s}} = R(x_c, x_o)$, where $\hat{\boldsymbol{s}}$ is a reconstruction of $\boldsymbol{s}$ with loss L_{recon} given by

$$L_{recon} = \left\| \boldsymbol{s} - \hat{\boldsymbol{s}} \right\|^2. \tag{2}$$

The BP sites are predicted by a network BP, the key component of the BP-GAN, that uses the context feature x_c corresponding to real sequences $\boldsymbol{s}$ to generate $\hat{\boldsymbol{b}} = BP(x_c)$. We denote a loss of BP by L_{BP} as

$$L_{BP} = -\frac{1}{N} \sum_{b_i \in \boldsymbol{b}, \hat{b}_i \in \hat{\boldsymbol{b}}} b_i \log(\hat{b}_i) + (1 - b_i) \log(1 - \hat{b}_i). \tag{3}$$

The generator G generates a sequence $\boldsymbol{s}_z = G(z)$ from $z \sim \mathcal{N}(0, 1)$. The BiGAN approach [14] demonstrated that a GAN can be improved by augmenting a feature encoder into the training process. As a result, the discriminator D of the BP-GAN additionally ingests the features x_c and x_o along with $\boldsymbol{b}$ to discriminate better between real and fake data. Thus, we rewrite L_{adv} in (1) as

$$\begin{aligned} -L_{adv} = {} & \mathbb{E}_{\boldsymbol{s} \sim P_s}\left[logD(\boldsymbol{s}, x_c, x_o, \boldsymbol{b}) \right] \\ & + \mathbb{E}_{z \sim \mathcal{N}(0,1)}\left[log(1 - D(\boldsymbol{s}_z, x_{z,c}, x_{z,o}, BP(x_{z,c}))) \right] \end{aligned} \tag{4}$$

where, $x_{z,c} = E_c(\boldsymbol{s}_z)$ and $x_{z,o} = E_o(\boldsymbol{s}_z)$.

C. SEQUENCE-BASED POSITIONAL SELF-ATTENTION

Stacked 1-D residual networks (ResNets) [32] and self-attention blocks feature largely in the network architecture of the BP-GAN. As discussed in Section II, self-attention aims to effectively learn long-term dependency of an input sequence. We used *Transformer* [24], a self-attention mechanism without using RNN by combining two Transformer blocks and three 1-D ResNet blocks, referring to a branchpoint-attention (BP-ATT) block as shown in Figure 2(a). The Transformer depicted in Figure 2(b) takes the input feature $x \in \mathbb{R}^{N \times d_k}$ corresponding to $\boldsymbol{s}$, where N is the sequence length of $\boldsymbol{s}$, and d_k is a dimension of x corresponding to an element of sequence $\boldsymbol{s}$ and set to 64. Given a query, Q, and a set of key-values, K–V with $Q, K, V \in \mathbb{R}^{N \times d_k}$ created from x, we evaluate the similarity between the key and query. Next, attention $Att \in \mathbb{R}^{N \times N}$ is given as a scaled dot product by

$$Att(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}} \right) V \tag{5}$$

As positional information is lost in self-attention, we apply a positional encoding operation to the input feature to translate it to a positional matrix $pe \in \mathbb{R}^{N \times d_k}$ as

$$\begin{aligned} pe_{pos,2i} &= \sin(pos/100000^{2i/d_k}), \\ pe_{pos,2i+1} &= \cos(pos/100000^{2i/d_k}) \end{aligned} \tag{6}$$

where, $pos \in \{1, \ldots, N\}$ is a sequence-positional index and $i \in \{1, \ldots, \frac{d_k}{2}\}$ is the dimensional index of x. We combine h results of scaled dot products to attend to different representation spaces jointly. The results are projected to a linear layer given by

$$\begin{aligned} M = {} & concat(Att(pe' W_1^Q, pe' W_1^K, pe' W_1^V), \ldots, \\ & Att(pe' W_h^Q, pe' W_h^K, pe' W_h^V)) W_o \end{aligned} \tag{7}$$

where W_i^* and W_o are trainable parameters, and $pe' = x + pe$. We then apply layer normalization to a feature M and add it to the input feature via a residual connection. The output of the Transformer block, $O = Norm(x + M)W_f$, is a projection

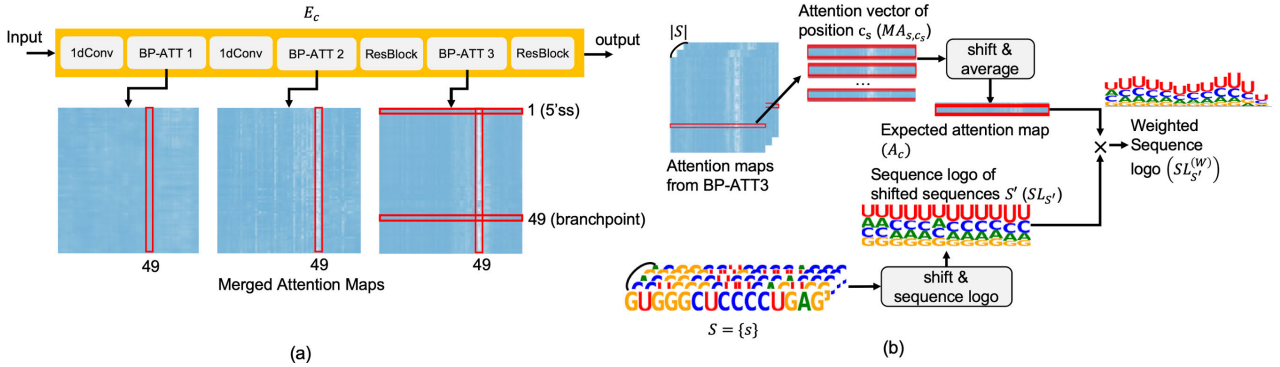

FIGURE 3. Interpretation of attention map for motif analysis: (a) An attention map merged from a BP-ATT block and (b) generation of weighted sequence logos of sequences. (Best viewed in color).

of the normalized sum $x + M$ to fully connected layers with trainable parameters W_f. The same procedure applies to the other BT-ATT blocks in Figure 3(a) except for the positional encoding. We used three BP-ATT blocks in E_C, two in E_o and G, and one in D. (See Appendix APPENDIX B for the full model details of the BP-GAN.)

D. MOTIF INTERPRETATION FROM ATTENTION

In addition to an increase in the accuracy, the BP-GAN provides interpretability on prediction decisions, using the self-attention mechanism as its key strength. The BP-GAN is specifically designed to identify motifs within an input sequence to predict their locations and attention intensity.

Given an intronic input sequence $\boldsymbol{s}$, the motif interpretation from the attention begins by summing h attention maps obtained from the last BP-ATT of E_c into a single attention map $MA_{\boldsymbol{s}} = \sum_{j=1}^{h} \text{softmax}(Q_j K_j^T/\sqrt{d_k}) \in \mathbb{R}^{N\times N}$, where Q_j and K_j are the key and value corresponding to the *Attention map*$_i$ as shown in Figure 2(b). It is likely that the last attention block delivers the most informative attention map. We denote the i–th row of $MA_{\boldsymbol{s}}$ by $MA_{\boldsymbol{s},i} \in \mathbb{R}^{1\times N}$. In the context of the Transformer model, $MA_{\boldsymbol{s},i}$ is an attention vector that represents the extent to which a sequence element $s_i \in \boldsymbol{s}$ attends to other nucleotides. Inversely, the vertical rectangles show the extent to which the associated nucleotides are being attended by others. Figure 3(a) shows an example of the three BP-ATT blocks in E_c for input $\boldsymbol{s}$ with a BP being s_{49}. The two horizontal rectangles in the figure depict attention vectors showing how s_1 (5' ss) and s_{49} (BP) attend to other nucleotides of the input. The attention intensity in the vertical rectangles for s_{49} is more distinguishable than other sites in the later attention maps. This observation explains why we chose the last BP-ATT for motif analysis, which is explained below. The example also suggests that s_{49} is an important clue for branch prediction as nucleotides in the input sequence attend to s_{49} the most.

Next, we define a shift function $sh(\boldsymbol{x}, k)$ that shifts all the elements in sequence $\boldsymbol{x}$ so that the k–th element is relocated to the center of $\boldsymbol{x}$ thereafter. Nucleotides outside the input sequence can be involved as a result of the shift operation. Given the set of test sequences $\boldsymbol{S} = \{\boldsymbol{s}\}$ under consideration, let $A_c \in \mathbb{R}^{1\times N}$ be an expected value of the shifted attention vectors, which is given by

$$A_c = \frac{1}{|\boldsymbol{S}|} \sum_{\boldsymbol{s}\in\boldsymbol{S}} sh(MA_{\boldsymbol{s},c_{\boldsymbol{s}}}, c_{\boldsymbol{s}}) \tag{8}$$

where $c_{\boldsymbol{s}}$ is a positional index to be the center of $\boldsymbol{s}$ and $|\boldsymbol{S}|$ is the cardinality of $\boldsymbol{S}$. However, each sequence may have different $c_{\boldsymbol{s}}$ when, for example, it refers to their BP sites. Given the predicted BPs $\hat{\boldsymbol{b}}_{\boldsymbol{s}}$ of sequence $\boldsymbol{s}$, we obtain an aggregated attention map A_c by considering $c_{\boldsymbol{s}} = \hat{\boldsymbol{b}}_{\boldsymbol{s}}$. This enables us to identify common motifs to which predicted BPs attended. Furthermore, if we seek motifs with respect to a specific position in the RNA sequence, e.g., the 3' sss, it is required to merely set all $c_{\boldsymbol{s}}$ identically to N.

We also define $SL_{\boldsymbol{S}} \in \mathbb{R}^{4\times N}$ as a sequence logo [33] corresponding to $\boldsymbol{S}$. Given a collection of aligned sequences, a sequence logo is a graphical representation of the sequence conservation of nucleotides in a strand of DNA/RNA. A sequence logo shows how frequently each type of nucleotide appears at each position along the horizontal axis as shown in Figure 3(b). The frequency of a type corresponds to the larger letter, depicting the consensus sequence and diversity of sequences. A sequence logo $SL_{\boldsymbol{S}'}$ is then created from the set of shifted sequences $\boldsymbol{S}' = \{sh(\boldsymbol{s}, c_{\boldsymbol{s}})\}$. Now, we define $SL^{(W)}_{\boldsymbol{S}'}$ as a weighted sequence logo using the mean attentions obtained from (8) corresponding to $\boldsymbol{S}'$. We now have

$$SL^{(W)}_{\boldsymbol{S}'} = SL_{\boldsymbol{S}'} \times \text{diag}(A_c) \tag{9}$$

where $\text{diag}(A_c)$ is a diagonal matrix with A_c being the diagonal entries.

To summarize, $SL^{(W)}_{\boldsymbol{S}'}$ visualizes the extent to which each of the nucleotides affects the BP sites using input sequences aligned to $c_{\boldsymbol{s}}$. In Section IV-C, we showed that the motifs discovered by the BP-GAN included to the consensus sequence[4]

[4] The consensus sequence is **yUnAy** in nucleotide codes, where **y** is **C** or **U** and **n** is any nucleotide in addition to **U** and **A**. Here, **A** represents a BP site.

and polypyrimidine tract that are well established in previous studies [6].

E. TRAINING THE BP-GAN END-TO-END

Owing to the biased distribution of BPs, there is class-imbalance that may cause overfitting. Therefore, we use a novel regularization based on triplet loss to learn a distance function from three samples namely, an anchor, a positive, and a negative [34]. This loss ensures that, in feature space, an anchor is always closer to a positive than a negative by a margin at least. Let us define $\boldsymbol{s}_{a,i}$ as an anchor having a BP at i, and $\boldsymbol{s}_{p,i}$ and $\boldsymbol{s}_{n,i}$ as a positive and negative, respectively. The triplet loss L_{tri} is given by

$$L_{tri} = \sum_{i=1}^{N} \left[\left\| x_{a,i} - x_{p,i} \right\|^2 - \left\| x_{a,i} - x_{n,i} \right\|^2 + \alpha \right]_+ \quad (10)$$

where, $[\cdot]_+ := max(0, \cdot)$ takes the positive component of its input, α is the margin, and $x_{*,i} = E_c(\boldsymbol{s}_{*,i})$.

A positive is a sequence with a different nucleotide at the same BP position. As the training proceeds, most of the negatives will be further apart from an anchor than the positives, making L_{tri} ineffective. Hard negative mining relieves this problem by favoring hard negatives over easy ones to evaluate L_{tri}. The selection of a hard negative to the anchor is based on the following assumptions: The features of sequences with a single BP will be similar as their BP sites are closer, and it will be harder to classify these sequences. Likewise, sequences with multiple BPs will be similar as they have more number of identical BPs. We considered sequences with a single BP in (10) for simplicity. (See Appendix APPENDIX A for more details on identifying hard negatives.)

During the training, not all the anchors may have their counterparts, positives, and hard negatives in a mini-batch. We augment auxiliary mini-batches into the training process to calculate L_{tri} in addition to normal mini-batches. In particular, we collect the triplets found in normal mini-batches and put them to an auxiliary mini-batch. When the auxiliary mini-batch has as many samples as a normal mini-batch, we perform a stochastic gradient descent on the auxiliary mini-batch to evaluate L_{tri}.

The entire BP-GAN is differentiable. Thus, the model can be trained end-to-end based on standard back propagation [35]. Combining (2), (3), (4), and (10), we rewrite the objective function in (1) by

$$\min_{\theta_D, \theta_{BP}, \theta_R} \max_{\theta_G} J(G, D, BP, R) = L_{BP}^{(nor)} + L_{recon}^{(nor)} + \lambda L_{adv} + \mathbb{1}_{aux} L_{tri}^{(aux)} \quad (11)$$

where λ is a coefficient of the adversarial loss, and the indicator function, $\mathbb{1}_{aux}$, on the right determines whether the training is on an auxiliary mini-batch. $L_{BP}^{(nor)}$ and $L_{recon}^{(nor)}$ are the expected losses corresponding to L_{BP} and L_{recon} for sequences in a normal mini-batch, while $L_{tri}^{(aux)}$ to L_{tri} in an auxiliary mini-batch. We used the GAN training procedure following [36], except for the auxiliary mini-batches. (See Appendix Algorithm 1.)

IV. RESULTS AND DISCUSSION

A. EXPERIMENTAL SETUP

1) IMPLEMENTATION DETAILS

We used the ADAM optimizer [39] with its parameters β_1 and β_2 set to 0.5 and 0.9, respectively, for the training. The size of a mini-batch was set to 512. The learning rate was set to 0.001. Margin α in (10) was set to 10. We set λ in (11) to 0.5. Auxiliary mini-batches were found to be augmented almost every five normal mini-batches during the training.

2) DATASET PREPROCESSING

We used two public datasets of RNA sequences extracted from chromosomes 1–22 and X in the reference human genome hg19 with a high confidence set of BP annotations, referred to as DS_P [5] and DS_M [4]. We set $N = 70$, taking the 3' ss and 69 of its upstream precedents to construct an input sequence as a 4×70 one-hot matrix. We split the datasets into three sets identical to the settings used in previous studies [3], [8], [12], [28], i.e., chromosome 1 as a test set, chromosomes 2, 3, 4, and 5 as a validation set, and the remaining as a training set. Table 1 provides the summary of the datasets regarding data splits when training BP-GAN and data imbalance ratio between BPs and other nucleotides. Figure 6 in Appendix shows a set of plots for the training and validation data with each of the datasets.

TABLE 1. Data splits for BP-GAN training and imbalance ratio between BP and other nucleotides on two datasets.

dataset	# test seq.	# val. seq.	# tr. seq.	BPs to others
DS_P [5]	7,202	13,494	46,042	1:45
DS_M [4]	4,306	7,093	25,693	1:39

B. QUANTITATIVE EVALUATION OF PREDICTION ACCURACY

1) OVERALL PERFORMANCE USING LARGE SCALE DATASETS

We used six performance metrics: the area under the receiver operating characteristic curve (auROC), area under precision-recall curve (auPRC), F-score, sensitivity (SE), specificity (SP), and recall at 1 (R@1). We set the decision threshold to 0.5 to measure the sensitivity, specificity, and F-score. A high sensitivity signifies that a model is likely to predict BPs with a higher confidence than the threshold. Similarly, a high specificity indicates that a model is effective to filter non-BPs. The auROC and auPRC are more informative with multiple thresholds than the basic metrics. We laid emphasis on auPRC over auROC with regard to class imbalance [40]. The ratio of the number of BPs to that of other nucleotides was approximately 1:45 and 1:39 in DS_M and DS_P, respectively. Another important measure namely, a large value in recall at K indicates that the K BP sites of highest probabilities as predicted by the model are highly likely to be the true BPs. This can improve the efficiency of biological experiments by reducing the number of BP candidates to be validated.

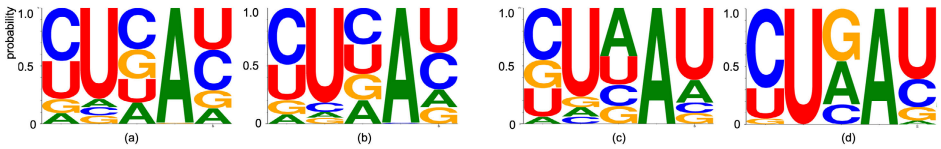

FIGURE 4. The portion of each type of nucleotides in the consensus motif `yUnAy` induced by the test sets of (a) DS_M and (b) DS_P, and sequences generated by $G(\cdot)$ of BP-GAN with the training sets of (c) DS_M and (d) DS_P respectively. (Best viewed in color).

TABLE 2. Evaluation of prediction accuracy using sensitivity (SE), specificity (SP), F-score, the area under receiver operating characteristic curve (auROC), area under precision-recall curve (auPRC), and recall at 1 (R@1) with the two datasets DS_P and DS_M. For ablation studies, we evaluated several variants of the BP-GAN in the lower half of the table by combining attention mechanisms (ATT) and regularization with triplet loss (REG) on top of default 1-D CNN and GAN. The best scores are highlighted in red.

Dataset	DS_P [5]						DS_M [4]					
Metric	SE	SP	F-score	auROC	auPRC	R@1	SE	SP	F-score	auROC	auPRC	R@1
SVM-BP [8]	-	-	-	-	-	-	-	-	-	0.632	0.408	0.471
branchpointer [12]	-	-	-	-	-	-	0.566	0.989	0.574	0.905	0.645	0.675
LaBranchoR [3]	0.482	0.996	0.585	0.972	0.662	0.755	0.448	0.996	0.560	0.963	0.687	0.743
Nazari *et al.* [28]	-	-	-	0.973	0.671	-	-	-	-	0.967	0.696	-
CNN	0.487	0.996	0.583	0.969	0.653	0.745	0.476	0.995	0.571	0.952	0.666	0.740
CNN+ATT	0.457	0.997	0.577	0.971	0.666	0.764	0.458	0.996	0.540	0.961	0.678	0.747
CNN+ATT+REG	0.488	0.996	0.592	0.972	0.666	0.762	0.410	0.997	0.536	0.968	0.688	0.753
GAN	0.511	0.992	0.603	0.965	0.632	0.747	0.466	0.995	0.559	0.950	0.679	0.744
GAN+ATT	0.501	0.996	0.595	0.970	0.661	0.754	0.481	0.995	0.577	0.970	0.703	0.753
GAN+ATT+REG (BP-GAN)	**0.512**	**0.996**	**0.604**	**0.971**	**0.680**	**0.766**	**0.489**	**0.995**	**0.579**	**0.963**	**0.710**	**0.756**

Table 2 compares the BP-GAN with four approaches namely, SVM-BP [8], branchpointer [12], LaBranchoR [3], and Nazari *et al.* [28], including two state-of-the-art studies, on the test sets from the two datasets. Not all performance measures are available for the methods, for example, no working implementations of SVM-BP [8] and Nazari *et al.* [28] exist so that we can use only the performance numbers reported in the literature; branchpointer [12] could be applied only to DS_M as it required extra information, that was not available, to use the new dataset.

The BP-GAN in the last row of the table outperforms SVM-BP and branchpointer except for sensitivity. In comparison with the best performing approaches, LaBranchoR [3] and Nazari *et al.* [28], the BP-GAN boots the performance over all metrics, except auROC on DS_P that is already saturated in all the approaches. The results indicate that the BP-GAN can be used effectively to identify real BP sites as well as filter sites that are non-BP. Notably, the performance gain BP-GAN over Nazari *et al.* [28] (0.009 on DS_P and 0.014 on DS_M) is identical at least or larger than those achieved by Nazari *et al.* [28] over LaBranchoR [3] (0.009 on DS_P and 0.009 on DS_M), in terms of auPRC.

In addition, we present additional evaluations on auPRC for a region [-45,-18] where most BP are located as shown in Table 3. The auPRC increased compared to the whole region evaluation in all the approaches. BP-GAN has the performance improvement similar to the case of whole region in Table 2 over LaBranchoR [3] and Nazari *et al.* [28]. We also present results with other chromosomes as test sets in Table 4 to show that the BP-GAN consistently outperforms other approaches.

TABLE 3. Additional evaluations on auPRC for BP-rich region [−45,−18]. The best scores are highlighted in red.

Dataset	DS_P [5]		DS_M [4]	
Metric	auPRC	auPRC [-45,-18]	auPRC	auPRC [-45,-18]
SVM-BP [8]	-	-	0.408	0.488
branchpointer [12]	-	-	0.645	0.712
LaBranchoR [3]	0.662	0.709	0.687	0.733
Nazari *et al.* [28]	0.671	0.718	0.696	0.742
BP-GAN	**0.680**	**0.728**	**0.710**	**0.753**

Figure 4 (a) and (b) depict two sequence logos corresponding to the test datasets of DS_P and DS_M that show the distribution of nucleotide types at BP predicted by our model. The figures illustrate that adenine ('A') is the dominant type of nucleotides at BP predicted and the sequences generated contained the consensus motif.

2) ABLATION STUDY

The pipeline of the BP-GAN combines several components: GAN, attention, and regularization with triple loss. We conducted an ablative study on these components and the results are shown in the bottom half of Table 2. A combination of all the components represents the BP-GAN. For comparison, we built CNN models by using the encoder with its network structure identical to E_c and E_o and the BP predictor BP; however, the encoder in the CNN models does not use the disentangled feature representation, unlike the BP-GAN. Moreover, only the cross-entropy loss is used for training the model without the adversarial term. The GAN models exhibited an improved performance in most of the metrics as compared to the CNN models with the same configurations of attention and regularization. In particular, the largest

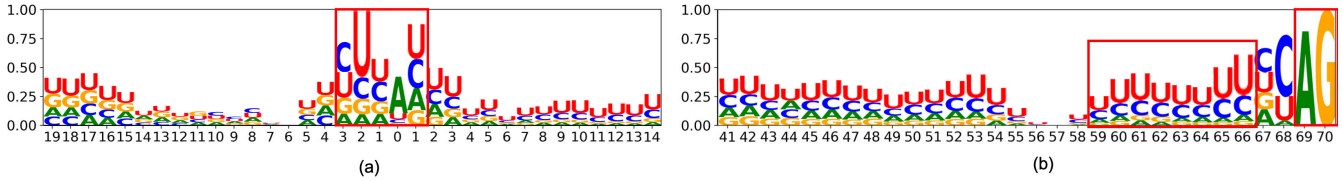

FIGURE 5. Interpretation of attention maps to generate weighted sequence logos. (a) An attention map merged from Transformer blocks was used to weight a sequence logo of sequences focusing on different areas such as (b) around a BP and the 3′ ss. (Best viewed in color).

TABLE 4. Performance evaluation on more chromosomes. The best scores are highlighted in red.

Chromosomes (# sequences)	chr2–3 (4456)			chr4–chr6 (4428)		
Metric	auROC	auPRC	R@1	auROC	auPRC	R@1
branchpointer [12]	0.914	0.548	0.622	0.911	0.556	0.623
LaBranchoR [3]	0.976	0.675	0.689	0.978	0.683	0.697
BP-GAN	**0.977**	**0.689**	**0.704**	**0.977**	**0.688**	**0.708**

Chromosomes (# sequences)	chr7–9 (3702)			chr10–11 (5216)		
Metric	auROC	auPRC	R@1	auROC	auPRC	R@1
branchpointer [12]	0.904	0.561	0.634	0.905	0.553	0.631
LaBranchoR [3]	0.976	0.694	0.725	0.978	0.694	0.718
BP-GAN	**0.976**	**0.701**	**0.723**	**0.976**	**0.695**	**0.717**

Chromosomes (# sequences)	chr12–chr15 (5216)			chr16–17 (4991)		
Metric	auROC	auPRC	R@1	auROC	auPRC	R@1
branchpointer [12]	0.906	0.562	0.651	0.890	0.563	0.651
LaBranchoR [3]	0.978	0.694	0.720	0.975	0.700	0.742
BP-GAN	**0.977**	**0.701**	**0.724**	**0.976**	**0.711**	**0.760**

Chromosomes (# sequences)	chr18–20 (3596)			chr21–22 + chrX (2615)		
Metric	auROC	auPRC	R@1	auROC	auPRC	R@1
branchpointer [12]	0.890	0.559	0.644	0.898	0.563	0.652
LaBranchoR [3]	0.977	0.706	0.744	0.977	0.703	0.734
BP-GAN	**0.976**	**0.716**	**0.752**	**0.970**	**0.700**	**0.731**

performance gain was achieved in auPRC. It was not clear whether the attention and triplet loss improved the sensitivity on top of GAN consistently. The attention improved auPRC significantly in the CNN as well as GAN models. The performance of the CNN models with attention was comparable with those of LaBranchoR [3] and Nazari *et al.* [28]. The regularization using triplet loss resulted in an additional, consistent improvement in the performance in most of the metrics on all the datasets. The gain with triplet loss is not as significant as that achieved from GAN or attention; however, the model is effectively generalized, as expected.

Figure 4(c) and (d) depict sequence logos showing the distribution of the nucleotide types for generated sequences by the generator in BP-GAN. We sampled 1000 RNA sequences from the generator $G(\cdot)$ of BP-GAN after training for 80 epochs until convergence. Then their BPs were predicted by $BP(\cdot)$. We repeated this sampling procedure 10 times and created the sequence logo corresponding to the 10,000 generated sequences. The sequence logos reveal the *consensus sequence* **yUnAy** clearly. This result shows that BP-GAN learnt the features for BP from input sequences successfully.

3) EVALUATION OF PREDICTION OF MULTIPLE BPs

As a significant portion of human introns contains multiple BPs [5], we conducted further investigations concerning their prediction. DS_P was the latest dataset with emphasis on multiple BPs hence, we compared our approach with LaBranchoR in recall for three splits of the test sets of DS_P corresponding to the number of BPs in a sequence K. We considered those cases with not more than three BPs: 4517 sequences of a single BP ($K = 1$), 1618 of dual BPs ($K = 2$), and 593 of triple BPs ($K = 3$). These occupied more than 95% of the dataset. Table 5 shows that the BP-GAN outperforms LaBranchoR in terms of prediction of multiple BPs. On the whole, the performances of the BP-GAN and LaBranchoR increased but converged at larger K. The BP-GAN, however, is much superior to LaBranchoR in smaller top-K recommendations. The results imply that the BP-GAN can be used to identify BPs from fewer proposals in a cost-effective manner.

C. INTERPRETATION AND VISUALIZATION OF ATTENTION FOR SEQUENCE MOTIF ANALYSIS

The last set of experiments dealt with the interpretation of the predictions for identification of motifs and its visualization as described in Section III-D. We selected two nucleotide sites namely, the BP and 3' ss. We considered the union of the test splits of DS_P and DS_M to be $\boldsymbol{S}$ in (8). We also set c_s to the BPs and 3' sss of the sequences in $\boldsymbol{S}$ and calculated their corresponding weighted sequence logos, which are depicted in Figure 5(a) and Figure 5(b), respectively. In Figure 5(a), we considered the BPs in $[-45, -15]$ to be investigated in those visualizations that accounted for almost 97% of the predicted sites.

In terms of relative distance from a BP, the interval $[-3, 2]$ of the weighted sequence logo in Figure 5(a) is highlighted by a dashed red rectangle. The motif observed in the interval is identical to the aforementioned *consensus sequence* in nucleotide codes, established based on previous studies [6]. Figure 5(b) highlights the interval $[40, 70]$ as the distance from the 3' ss in the weighted sequence logo. Similar to the previous case, another well-known sequence motif is marked by the wide red rectangle. This is the *polypyrimidine tract* consisting of rich `C` and `U`. In addition, the BP-GAN affirms that the motif '`AG`' at the 3' ss is crucial for BP prediction, which is depicted by the narrow rectangle on the right side in Figure 5(b). Note that such the existing approaches are not able to identify such sequence motifs at distal and variable

TABLE 5. Comparison of BP-GAN with LaBranchoR [3] in terms of multi-BP prediction in recall at K with three sequence sets of identical BPs. The best scores are highlighted in red.

# BPs	1						2					3			
R@	1	2	3	4	5	6	2	3	4	5	6	3	4	5	6
LaBranchoR [3]	0.686	0.824	0.888	0.913	0.932	0.943	0.647	0.774	0.837	0.880	0.899	0.667	0.764	0.823	0.861
BP-GAN	**0.713**	**0.844**	**0.895**	**0.921**	**0.937**	**0.950**	**0.671**	**0.782**	**0.852**	**0.883**	**0.904**	**0.689**	**0.775**	**0.831**	**0.867**

TABLE 6. Comparison of BP-GAN and Nazari *et al.* [28] in detecting sequence variants on two datasets.

dataset	ClinVar [37]																		EpilepsyGene [38]										
Distance from BP	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	-3	-2	-1	0	1	2	3	4	5	6	7
# var. known	-	-	-	-	-	1	-	-	-	3	-	2	-	3	3	3	2	-	1	1	-	-	-	-	1	-	-	-	1	1
Nazari *et al.* [28]	1	-	-	-	-	1	-	-	-	-	-	2	2	-	-	1	-	1	-	-	-	-	-	-	1	-	-	-	-	-
BP-GAN	**-**	**-**	**-**	**-**	**-**	**1**	**-**	**-**	**-**	**1**	**-**	**2**	**1**	**1**	**1**	**1**	**-**	**-**	**-**	**1**	**-**	**-**	**-**	**-**	**1**	**-**	**-**	**-**	**1**	**-**

locations from BP sites. The results of the interpretation from attention demonstrate that, without any prior biological knowledge, the BP-GAN gains meaningful insights in terms of prediction of sequence motifs. The sequence logo analysis is an added advantage of the BP-GAN to identify novel sequence motifs for providing human-interpretable reasoning on its decisions, leading to a better understanding of RNA splicing.

The proposed method of interpretation using the BP-GAN is general as the effect of any region of RNA sequences on branch prediction can be identified directly. For example, if we enlarge an input sequence, distal motifs from a BP can be identified, if they exist. To the best of our knowledge, none of the previous literature addressed this aspect of BP using deep neural networks. An investigation in this area may be pursued in future.

D. EVALUATION ON VARIANTS

We investigated the effect of sequence variants on the model prediction. We used two public datasets of sequence variants, ClinVar [37] and EpilepsyGene [38], While ClinVar contains sequence variants and corresponding phenotype annotations for a wide variety of diseases, EpilepsyGene focuses on epilepsy disease. For comparing with the state-of-the-art work, we performed the evaluation similarly to what was done in Nazari *et al*. [28]. In particular, we selected 18 variants from ClinVar that are known to cause non-functional protein and aligned the variants to sequences in DS_P to constitute a candidate subset for the evaluation. Then, we modified each of the nucleotides within [-9,9] in distance from the predicted BP site and repeatedly applied the variants to BP-GAN to see if the prediction of BP location changes. Table 6 show the number of variants that changed BP predictions of the two methods and match the associated variants in ClinVar. Overall, BP-GAN has better agreement to the variants in ClinVar compared to that of Nazari *et al*. [28]. Especially, BP-GAN achieved 0.38 in sensitivity and 0.88 in precision while 0.22 and 0.57 with Nazari *et al*. [28] respectively. This reveals that BP-GAN detected more true variants and less false positive ones favorable against the existing approaches.

In addition, we performed the similar evaluation on the EpilepsyGene dataset. Even though the number of variants are smaller than that the case of ClinVar, the similar performance gain is observed.

TABLE 7. Architecture of the generator (*G*).

Operation	Input shape	Kernel size	Output shape
z	(k,35)	-	(k,35)
fc1&reshape	(k,35)	(35, 560)	(k,8,70)
batchNorm & ReLU	(k,8,70)	-	(k,8,70)
transpose	(k,8,70)	(-1,-2)	(k,70,8)
BP-ATT	**(k,70,8)**	**-**	**(k,70,8)**
transpose	(k,70,8)	(-1,-2)	(k,8,70)
ResBlock1	(k,8,70)	(5,8,8) $\times 2 \times 3$	(k,8,70)
conv1d	(k,8,70)	(1,8,64)	(k,64,70)
batchNorm & ReLU	(k,64,70)	-	(k,64,70)
transpose	(k,64,70)	(-1,-2)	(k,70,64)
BP-ATT	**(k,70,64)**	**-**	**(k,70,64)**
transpose	(k,70,64)	(-1,-2)	(k,64,70)
ResBlock2	(k,64,70)	(5,64,64) $\times 2 \times 3$	(k,64,70)
conv1d	(k,64,70)	(1,64,4)	(k,4,70)
softmax	(k,4,70)	dim = 1	(k,4,70)

V. CONCLUSIONS

We introduced the BP-GAN, a novel integration of attention and triplet loss on top of a GAN-based framework for human BP prediction from RNA sequences. Our approach used a disentangled representation of RNA sequences with a GAN to learn BP prediction-aware features. Attention learned intra-sequence dependencies, delivering improved prediction accuracy and interpretability on decisions. Triplet loss addressed the issue of class-imbalance using hard negative mining. All these techniques contributed to exhibiting a state-of-the-art-performance on the two large-scale datasets. The sequence motifs identified from the attention layers of the BP-GAN agreed with the consensus sequence, polypyrimidine tract, and 3' ss that are known to essentially affect BP sites during splicing. This enabled us to gain biologically meaningful insights to explain the predictions. To the best of our knowledge, this is the first attempt in applying a GAN-based model to a discriminative task in gene expression. Moreover, the BP-GAN can easily be extended to several other similar tasks.

APPENDIX A HARD NEGATIVE MINING FOR TRIPLET LOSS

Consider a sequence $\boldsymbol{s}$ with $\boldsymbol{b_s} \in \{0, 1\}^N$ being its multiple BPs. If we define $\boldsymbol{s}_{a,i}$ as an anchor with a BP at i, a hard

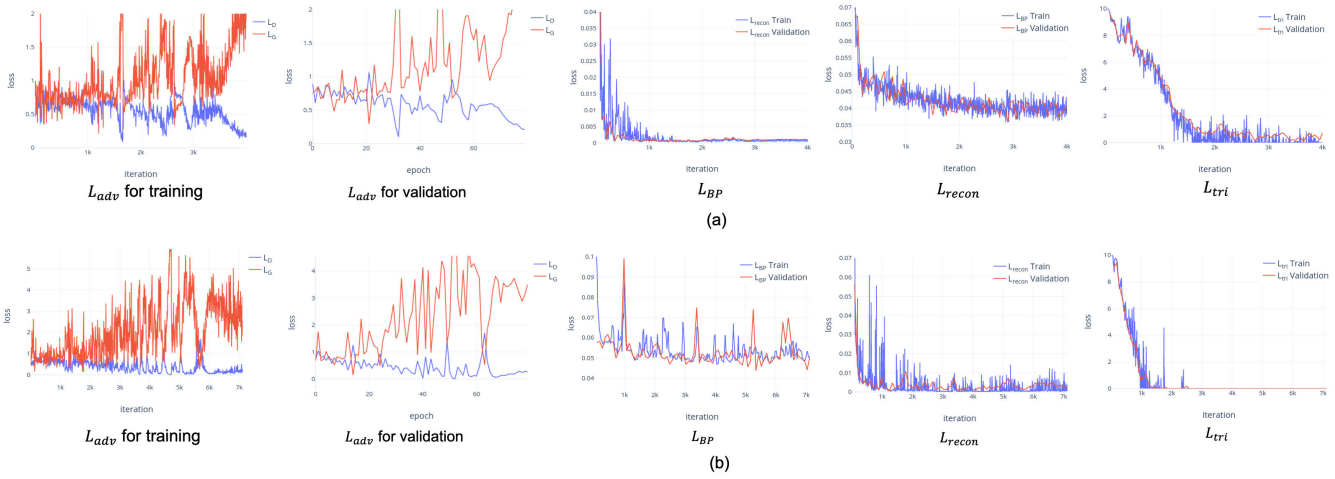

FIGURE 6. Training and validation plots for training BP-GAN with (a) DS_P and (b) DS_M. (Best viewed in color).

TABLE 8. Architecture of the discriminator (D).

Operation	Input shape	Kernel size	Output shape
S	(k,4,70)	-	(k,4,70)
conv1d seq	(k,4,70)	(5,4,32)	(k,32,70)
batchNorm + LeakyReLU	(k,32,70)	-	(k,32,70)
ResBlock seq	(k,32,70)	(5,32,32)× 2 × 2	(k,32,70) to concat
B	(k,1,70)	-	(k,1,70)
conv1d bp1	(k,1,70)	(5,1,8)	(k,8,70)
batchNorm + LeakyReLU	(k,8,70)	-	(k,8,70)
conv1d bp2	(k,8,70)	(1,8,32)	(k,32,70)
batchNorm + LeakyReLU	(k,32,70)	-	(k,32,70)
ResBlock bp	(k,32,70)	(5,32,32) × 2	(k,32,70) to concat
x_c, x_o	(k,64,70), (k,64,70)	-	(k,64,70), (k,64,70) to concat
concat seq, bp, x_c, x_o	(k,32,70), (k,32,70), (k,64,70), (k,64,70)	dim=1	(k,192,70)
conv1d	(k,192,70)	(1,192,64)	(k,64,70)
batchNorm +LeakyReLU	(k,64,70)	-	(k,64,70)
transpose	(k,64,70)	(-1,-2)	(k,70,64)
BP-ATT	**(k,70,64)**	-	**(k,70,64)**
transpose	(k,70,64)	(-1,-2)	(k,64,70)
ResBlock	(k,64,70)	(5,64,64) × 2 × 3	(k,64,70)
conv1d	(k,64,70)	(1,64,16)	(k,16,70)
batchNorm +LeakyReLU	(k,16,70)	-	(k,16,70)
resize	(k,16,70)	-	(k,1120)
fc sigmoid	(k,1120)	(1120,1)	(k)

negative $\boldsymbol{s}_{n,i}$ to $\boldsymbol{s}_{a,i}$ is given by

$$\boldsymbol{s}_{n,i} = \begin{cases} \boldsymbol{s}_j, & \text{if } i \neq j \text{ and } |i-j| \leq K \text{ and } ||\boldsymbol{b}_{\boldsymbol{s}_{a,i}}||_1 = 1 \\ \boldsymbol{s}'_i, & \text{if } \boldsymbol{b}_{\boldsymbol{s}_{a,i}} \cdot \boldsymbol{b}_{\boldsymbol{s}'_i} < ||\boldsymbol{b}_{\boldsymbol{s}_{a,i}}||_1 \text{ and } ||\boldsymbol{b}_{\boldsymbol{s}'_i}||_1 > 1 \end{cases} \quad (12)$$

where, $||.||_1$ is the L_1 norm. The upper case in (12) corresponds to sequences of a single BP whereas the lower case accounts for those of multiple BPs. We set K to 64 in the experiments.

TABLE 9. Architecture of the sequence encoders (E_c and E_o). The entries applied only to E_c or E_o are specified in parentheses.

Operation	Input shape	Kernel size	Output shape
S	(k,4,70)	-	(k,4,70)
conv1d share	(k,4,70)	(5,4,128)	(k,128,70)
batchNorm ReLU	(k,128,70)	-	(k,128,70)
transpose	(k,128,70)	(-1,-2)	(k,70,128)
BP-ATT	**(k,70,128)**	-	**(k,70,128)**
transpose	(k,70,128)	(-1,-2)	(k,128,70)
ResBlock share	(k,128,70)	(5,128,128) ×2 × 3	(k,128,70) to conv1d (E_c) and conv1d (E_o)
conv1d (E_c)	(k,128,70)	(1,128,64)	(k,64,70)
batchNorm ReLU	(k,64,70)	-	(k,64,70)
transpose	(k,64,70)	(-1,-2)	(k,70,64)
BP-ATT (E_c)	**(k,70,64)**	-	**(k,70,64)**
transpose	(k,70,64)	(-1,-2)	(k,64,70)
Resblock (E_c)	(k,64,70)	(5,64,64) × 2 × 3	(k,64,70)
transpose	(k,64,70)	(-1,-2)	(k,70,64)
BP-ATT (E_c)	**(k,70,64)**	-	**(k,70,64)**
transpose	(k,70,64)	(-1,-2)	(k,64,70)
ResBlock (E_c)	(k,64,70)	(5,64,64) × 2 × 3	(k,64,70)
conv1d (E_o)	(k,128,70)	(1,128,64)	(k,64,70)
batchNorm ReLU	(k,64,70)	-	(k,64,70)
transpose	(k,64,70)	(-1,-2)	(k,70,64)
BP-ATT (E_o)	**(k,70,64)**	-	**(k,70,64)**
transpose	(k,70,64)	(-1,-2)	(k,64,70)
Resblock (E_o)	(k,64,70)	(5,64,64) × 2 × 3	(k,64,70)

TABLE 10. Architecture of the BP predictor (BP).

Operation	Input shape	Kernel size	Output shape
E_c	(k,64,70)	-	(k,64,70)
cov1d	(k,64,70)	(1,64,1)	(k,1,70)
sigmoid	(k,1,70)	-	(k,1,70)

APPENDIX B ARCHITECTURE DESCRIPTION

Table 7–11 describes the entire architecture of the BP-GAN.

APPENDIX C PSEUDO CODE FOR TRAINING BP-GAN END-TO-END

Algorithm 1 shows the training process of the BP-GAN.

TABLE 11. Architecture of the sequence decoder (R).

Operation	Input shape	Kernel size	Output shape
E_c	(k,64,70)	-	(k,64,70)
E_o	(k,64,70)	-	(k,64,70)
concat E_c, E_o	(k,64,70),(k,64,70)	dim=1	(k,128,70)
conv1d	(k,128,70)	(1,128,64)	(k,64,70)
batchNorm & ReLU	(k,64,70)	-	(k,64,70)
ResBlock	(k,64,70)	(5,64) × 2 × 2	(k,64,70)
conv1d	(k,64,70)	(1,64,16)	(k,16,70)
batchNorm & ReLU	(k,16,70)	-	(k,16,70)
conv1d	(k,16,70)	(1,16,4)	(k,4,70)
batchNorm & ReLU	(k,4,70)	-	(k,4,70)
softmax	(k,4,70)	dim = 1	(k,4,70)

Algorithm 1 Training Process of BP-GAN

for number of training iterations **do**
  for m in 1:*MinibatchSize* **do**
    Sample a random noise vector $z \sim \mathcal{N}(0, 1)$
    $s_z \leftarrow G(z)$
    $x_{z,c} \leftarrow E_c(s_z)$
    $x_{z,o} \leftarrow E_o(s_z)$
    Predict branchpoints $BP(x_{z,c})$.
    Add a quadruple $(s_z, x_{z,c}, x_{z,o}, BP(x_{z,c}))$ to the current normal mini-batch with the label 0 as fake
  end for

$\{\theta_G, \theta_{E_c}, \theta_{E_o}, \theta_{BP}\} \leftarrow \{\theta_G, \theta_{E_c}, \theta_{E_o}, \theta_{BP}\} + \nabla_{\{\theta_G, \theta_{E_c}, \theta_{E_o}, \theta_{BP}\}} L_{adv}$

  for m in 1:*MinibatchSize* **do**
    Sample a sequence $\boldsymbol{s} \sim P_s$
    $x_c \leftarrow E_c(S)$
    $x_o \leftarrow E_o(S)$
    Add a quadruple $(\boldsymbol{s}, x_c, x_o, \boldsymbol{b})$ to the current normal mini-batch, assigning the label 1 as real
    $\hat{\boldsymbol{b}} \leftarrow BP(x_c)$
    $\hat{\boldsymbol{s}} \leftarrow R(x_c, x_o)$
  end for

$\theta_D \leftarrow \theta_D - \nabla_{\theta_D} L_{adv}$
$\{\theta_{E_c}, \theta_{BP}\} \leftarrow \{\theta_{E_c}, \theta_{BP}\} - \nabla_{\{\theta_{E_c}, \theta_{BP}\}} L_{BP}^{(nor)}$
$\{\theta_{E_c}, \theta_{E_o}, \theta_R\} \leftarrow \{\theta_{E_c}, \theta_{E_o}, \theta_R\} - \nabla_{\{\theta_{E_c}, \theta_{E_o}, \theta_R\}} L_{recon}^{(nor)}$

  for m in 1:*AuxiliaryMinibatchSize* **do**
    Sample an anchor $\boldsymbol{s}_a \sim P_s$
    Sample a positive $\boldsymbol{s}_p \sim P_s$
    Sample a hard negative $\boldsymbol{s}_n \sim P_s$
    $x_{a,c} \leftarrow E_c(\boldsymbol{s}_a)$
    $x_{p,c} \leftarrow E_c(\boldsymbol{s}_p)$
    $x_{n,c} \leftarrow E_c(\boldsymbol{s}_n)$
  end for

$\theta_{E_c} \leftarrow \theta_{E_c} - \nabla_{\theta_{E_c}} L_{tri}^{(aux)}$
end for

Figure 6 shows two sets of plots for training BP-GAN with the training and validation data splits corresponding to the terms in (11) for DS_P and DS_M.

REFERENCES

[1] R. Singh, J. Lanchantin, A. Sekhon, and Y. Qi, "Attend and predict: Understanding gene regulation by selective attention on chromatin," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6785–6795.

[2] K. H. Lim, L. Ferraris, M. E. Filloux, B. J. Raphael, and W. G. Fairbrother, "Using positional distribution to identify splicing elements and predict pre-mRNA processing defects in human genes," *Proc. Nat. Acad. Sci. USA*, vol. 108, no. 27, pp. 11093–11098, 2011.

[3] J. M. Paggi and G. Bejerano, "A sequence-based, deep learning model accurately predicts RNA splicing branchpoints," *RNA*, vol. 24, no. 12, pp. 1647–1658, Dec. 2018.

[4] T. R. Mercer, M. B. Clark, S. B. Andersen, M. E. Brunck, W. Haerty, J. Crawford, R. J. Taft, L. K. Nielsen, M. E. Dinger, and J. S. Mattick, "Genome-wide discovery of human splicing branchpoints," *Genome Res.*, vol. 25, no. 2, pp. 290–303, Feb. 2015.

[5] J. M. B. Pineda and R. K. Bradley, "Most human introns are recognized via multiple and tissue-specific branchpoints," *Genes Develop.*, vol. 32, nos. 7–8, pp. 577–591, Apr. 2018.

[6] K. Gao, A. Masuda, T. Matsuura, and K. Ohno, "Human branch point consensus sequence is yUnAy," *Nucleic Acids Res.*, vol. 36, no. 7, pp. 2257–2267, Apr. 2008, doi: 10.1093/nar/gkn073.

[7] A. J. Taggart, C.-L. Lin, B. Shrestha, C. Heintzelman, S. Kim, and W. G. Fairbrother, "Large-scale analysis of branchpoint usage across species and cell lines," *Genome Res.*, vol. 27, no. 4, pp. 639–649, Apr. 2017.

[8] A. Corvelo, M. Hallegger, C. W. J. Smith, and E. Eyras, "Genome-wide association between branch point properties and alternative splicing," *PLoS Comput. Biol.*, vol. 6, no. 11, Nov. 2010, Art. no. e1001016.

[9] W. Zhang, X. Zhu, Y. Fu, J. Tsuji, and Z. Weng, "Predicting human splicing branchpoints by combining sequence-derived features and multi-label learning methods," *BMC Bioinf.*, vol. 18, no. S13, p. 464, Nov. 2017.

[10] J. Wen, J. Wang, Q. Zhang, and D. Guo, "A heuristic model for computational prediction of human branch point sequence," *BMC Bioinf.*, vol. 18, no. 1, p. 459, Dec. 2017.

[11] Q. Zhang, X. Fan, Y. Wang, M.-A. Sun, J. Shao, and D. Guo, "BPP: A sequence-based algorithm for branch point prediction," *Bioinformatics*, vol. 33, no. 20, pp. 3166–3172, Oct. 2017.

[12] B. Signal, B. S. Gloss, M. E. Dinger, and T. R. Mercer, "Machine learning annotation of human branchpoints," *Bioinformatics*, vol. 34, no. 6, pp. 920–927, Mar. 2018.

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[14] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," 2016, *arXiv:1605.09782*. [Online]. Available: http://arxiv.org/abs/1605.09782

[15] N. Killoran, L. J. Lee, A. Delong, D. Duvenaud, and B. J. Frey, "Generating and designing DNA with deep generative models," 2017, *arXiv:1712.06148*. [Online]. Available: http://arxiv.org/abs/1712.06148

[16] N. Anand and P. Huang, "Generative modeling for protein structures," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7494–7505.

[17] A. Gupta and J. Zou, "Feedback GAN for DNA optimizes protein functions," *Nature Mach. Intell.*, vol. 1, no. 2, p. 105, 2019.

[18] P. Chhibbar and A. Joshi, "Generating protein sequences from antibiotic resistance genes data using generative adversarial networks," 2019, *arXiv:1904.13240*. [Online]. Available: http://arxiv.org/abs/1904.13240

[19] Y. Wang, H. Wang, L. Liu, and X. Wang, "Synthetic promoter design in escherichia coli based on generative adversarial network," *bioRxiv*, 2019, Art. no. 563775.

[20] A. Ghahramani, F. M. Watt, and N. M. Luscombe, "Generative adversarial networks uncover epidermal regulators and predict single cell perturbations," *bioRxiv*, 2018, Art. no. 262501.

[21] R. Lopez, J. Regier, M. B. Cole, M. I. Jordan, and N. Yosef, "Deep generative modeling for single-cell transcriptomics," *Nature Methods*, vol. 15, no. 12, p. 1053, 2018.

[22] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015, *arXiv:1508.04025*. [Online]. Available: http://arxiv.org/abs/1508.04025

[23] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," 2016, *arXiv:1601.06733*. [Online]. Available: http://arxiv.org/abs/1601.06733

[24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[25] J. Lanchantin, R. Singh, B. Wang, and Y. Qi, "Deep motif dashboard: Visualizing and understanding genomic sequences using deep neural networks," in *Proc. Pacific Symp. Biocomput.*, 2017, pp. 254–265.
[26] S. Park, S. Min, H.-S. Choi, and S. Yoon, "Deep recurrent neural network-based identification of precursor micrornas," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2891–2900.
[27] V. Dean, A. Delong, and B. Frey, "Deep learning for branch point selection in RNA splicing," in *Proc. NIPS Workshop Mach. Learn. Comput. Biol. (MLCB)*, 2016, pp. 1–5.
[28] I. Nazari, H. Tayara, and K. T. Chong, "Branch point selection in RNA splicing using deep learning," *IEEE Access*, vol. 7, pp. 1800–1807, 2019.
[29] U. Lodish, H. Lodish, J. Darnell, A. Berk, C. Kaiser, C. Kaiser, U. Kaiser, M. Krieger, M. P. Scott, and A. Bretscher, H. Ploegh, P. Matsudaira, H. Ploegh, and P. Matsudaira, *Molecular Cell Biology*. New York, NY, USA: W. H. Freeman, 2008. [Online]. Available: https://books.google.co.kr/books?id=K3JbjG1JiUMC
[30] J. Zuallaert, F. Godin, M. Kim, A. Soete, Y. Saeys, and W. De Neve, "SpliceRover: Interpretable convolutional neural networks for improved splice site prediction," *Bioinformatics*, vol. 34, no. 24, pp. 4180–4188, Dec. 2018.
[31] R. Leman *et al.*, "Assessment of branch point prediction tools to predict physiological branch points and their alteration by variants," *BMC Genomics*, vol. 21, no. 1, p. 86, Dec. 2020, doi: 10.1186/s12864-020-6484-5.
[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[33] T. D. Schneider and R. M. Stephens, "Sequence logos: A new way to display consensus sequences," *Nucleic Acids Res.*, vol. 18, no. 20, pp. 6097–6100, 1990.
[34] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 815–823.
[35] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[36] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*. [Online]. Available: http://arxiv.org/abs/1511.06434
[37] M. J. Landrum, J. M. Lee, G. R. Riley, W. Jang, W. S. Rubinstein, D. M. Church, and D. R. Maglott, "ClinVar: Public archive of relationships among sequence variation and human phenotype," *Nucleic Acids Res.*, vol. 42, no. D1, pp. D980–D985, Jan. 2014.
[38] X. Ran, J. Li, Q. Shao, H. Chen, Z. Lin, Z. S. Sun, and J. Wu, "EpilepsyGene: A genetic resource for genes and mutations related to epilepsy," *Nucleic Acids Res.*, vol. 43, no. D1, pp. D893–D899, Jan. 2015.
[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980
[40] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 233–240.
[41] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2172–2180.
[42] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, "Visualizing deep neural network decisions: Prediction difference analysis," *CoRR*, 2017. [Online]. Available: http://arxiv.org/abs/1702.04595
[43] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.
[44] A. Dutta, T. Dubey, K. K. Singh, and A. Anand, "SpliceVec: Distributed feature representations for splice junction prediction," *Comput. Biol. Chem.*, vol. 74, pp. 434–441, Jun. 2018.
[45] A. Jha, M. R. Gazzara, and Y. Barash, "Integrative deep models for alternative splicing," *Bioinformatics*, vol. 33, no. 14, pp. i274–i282, Jul. 2017.
[46] M. Oubounyt, Z. Louadi, H. Tayara, and K. T. Chong, "Deep learning models based on distributed feature representations for alternative splicing prediction," *IEEE Access*, vol. 6, pp. 58826–58834, 2018.
[47] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," 2018, *arXiv:1805.08318*. [Online]. Available: http://arxiv.org/abs/1805.08318
[48] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7794–7803.
[49] G. King and L. Zeng, "Logistic regression in rare events data," *Political Anal.*, vol. 9, no. 2, pp. 137–163, 2001.
[50] H. W. Lilliefors, "On the kolmogorov-smirnov test for normality with mean and variance unknown," *J. Amer. Stat. Assoc.*, vol. 62, no. 318, pp. 399–402, Jun. 1967.

HYEONSEOK LEE received the B.S. and M.S. degrees in computer science and engineering from Jeonbuk National University, Jeonju, South Korea, in 2018 and 2020, respectively, where he is currently pursuing the Ph.D. degree. His main research interests include explainable AI and video understanding.

SANGWOO YEOM received the B.S. degree in statistics and the M.S. degree in computer science and engineering from Jeonbuk National University, Jeonju, South Korea, in 2017 and 2019, respectively. His main research interest includes the application of machine learning to various topics in genomics.

SUNGCHAN KIM received the B.S. degree in material science and engineering, the M.S. degree in computer engineering, and the Ph.D. degree in electrical engineering and computer science from Seoul National University, Seoul, South Korea, in 1998, 2000, and 2005, respectively. He is currently a Professor with the Division of Computer Science and Engineering, Jeonbuk National University, South Korea. His research interests include various topics in machine learning and computer vision.