Supplementary Materials for
Inference of splicing motifs through visualization of recurrent networks

1 Background

1.1 Biological preliminaries

Genes, in eukaryotes, comprise of alternating regions called *exons* and *introns*. During transcription, a process called splicing removes the introns and ligates the exons with the help of a complex molecular machine called *spliceosome*. Splicing occurs at exon-intron (*donor site*) and intron-exon (*acceptor site*) boundaries called *splice sites* or *splice junctions*.

However, more than 90% of multi-exon genes undergo *alternative splicing* (AS) wherein an intron may be retained and an exon may be skipped in a pre-mRNA transcript during splicing [1]. AS results in the formation of multiple protein isoforms from a single gene, thereby increasing the informational diversity and functional capacity of a gene. Accurate identification of splice junctions is, therefore, important for unfolding the gene structure and expression along with providing meaningful insights into its role in alternative splicing.

The spliceosome, during splicing, identifies the splice sites based on splicing consensus residing in the vicinity of the splice junctions. Splice junctions generally comprise of the canonical splicing pattern GT and AG at the donor and acceptor sites respectively. Apart from this, the consensus sequence comprises of an extended donor site consensus 9-mer $[AC]AGGTRAGT$ and an extended acceptor site consensus 15-mer $Y_{10}NCAGG$ containing a polypyrimidine tract (PY-tract) preceding the acceptor site [2]. There is also a very weak branch site consensus $CTRAY$ located approximately 30 nucleotides (nt) upstream from the acceptor site. However, there are evidences of existence of non-canonical splice junctions that do not comply with the above consensus.

1.2 Preliminaries on model architecture

Recurrent neural networks (RNN) An RNN is a class of artificial neural network that comprises of a hidden feedback unit which gets repeated each time an input is fed into the network. For a given input sequence $x = (x_1, x_2, ..., x_T)$, any x_t where $t \in 1, 2, ..., T$, can be considered as a time step. An RNN can be unrolled along time steps, as shown in Figure 1, to form a directed graph along the input sequence. This allows it to exhibit dynamic temporal behavior for a sequence.

The weights W_{xh}, W_{hh} and W_{hy}, of the input, hidden and output layers respectively, are the same across every time step. This suggests that an RNN performs the same task on every element of an input sequence, with dependency on the previous time step. The hidden state h_t at time step t of an RNN behaves as the memory of the network that is computed based on previous hidden state h_{t-1} and current input state x_t. This can be represented by the following recurrence formula:

$$h_t = f(h_{t-1}, x_t)$$

where f is any non-linearity function of the hidden layer. This can be further expanded as:

$$h_t = f(W_{hh}h_{t-1} + W_{xh}x_t)$$

The output at time step t is computed by:

$$y_t = W_{hy}h_t$$

Long short-term memory (LSTM) units Training a vanilla RNN involves updating the input, hidden and output weights, using backpropagation through time, to reduce the total error of the network. Weight of each time step of an unrolled RNN is updated in proportion to the derivative of the error with respect to

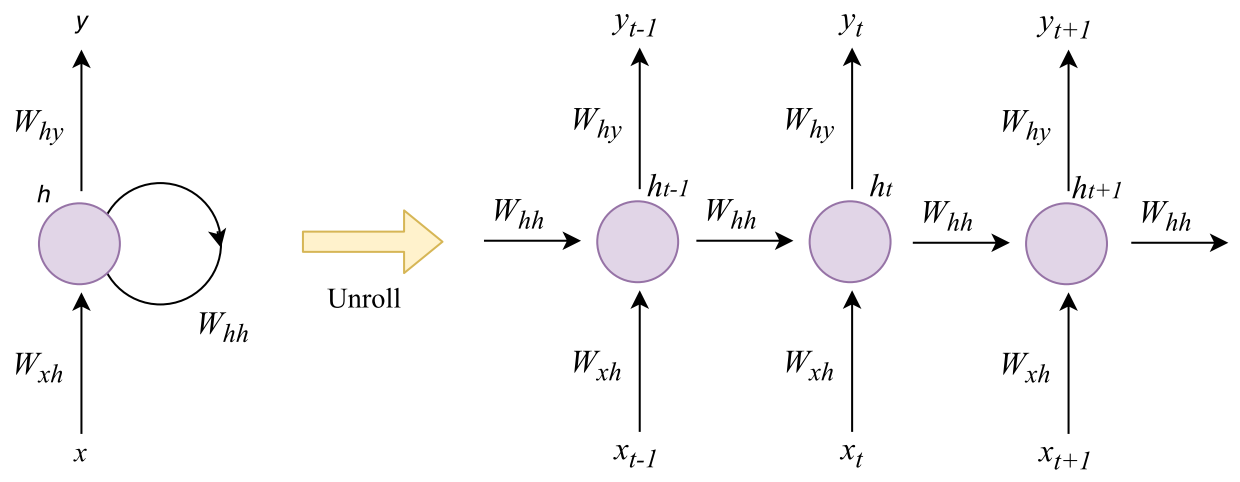


Figure 1: An unrolled RNN.

the weight. As we move along the time steps, the backward flow of the gradient results in either exploding or diminishing of the gradient exponentially. This occurs because a portion of the weight matrix gets multiplied by itself at each step of backpropagation. Hence, the vanilla RNN architecture was not capable of accessing long-term dependencies.

LSTM units are a type of building blocks, for the layers of an RNN, that are capable of memorizing long-term dependencies. LSTM units behave like memory cells comprising of input, output and forget gates. Figure 2 illustrates a single LSTM unit. The backpropagation passes through only the cell state which is controlled by the three gates. The activation vectors i_t, o_t, f_t, and c'_t, for the input gate, output gate, forget gate and candidate cell state at time step t, can be represented by the following equations:

$$i_t = \sigma(W_{hi}h_{t-1} + W_{xi}x_t)$$

$$o_t = \sigma(W_{ho}h_{t-1} + W_{xo}x_t)$$

$$f_t = \sigma(W_{hf}h_{t-1} + W_{xf}x_t)$$

$$c'_t = tanh(W_{hc}h_{t-1} + W_{xc}x_t)$$

The actual cell state value (C_t) at time step t is computed by:

$$C_t = f_tC_{t-1} + i_tc'_t$$

Finally, the output (h_t) of the unit at time step t is given by:

$$h_t = o_t tanh(C_t)$$

Bidirectional long short-term memory (BLSTM) networks A limitation of traditional RNN is that it makes decisions based on previous context only. However, language translation and speech recognition systems, where whole input is transcribed at once, exploiting future context is expected to improve predictive performance.

A BLSTM network is the bidirectional variant of an RNN that employs an LSTM unit in the hidden layer. Figure 3 shows a BLSTM network. It comprises of two identical hidden LSTM layers. One of the layers is fed with the input as-is whereas the other layer is fed with the reversed copy of the same input. Output from both the hidden layers is then combined and sent to the output layer. The forward hidden sequence ($\overrightarrow{h}$), the backward hidden sequence ($\overleftarrow{h}$), and the output y_t at time step t is given by:

$$\overrightarrow{h}_t = f(W_{x\overrightarrow{h}}x_t + W_{\overrightarrow{h}\overrightarrow{h}}\overrightarrow{h}_{t-1})$$

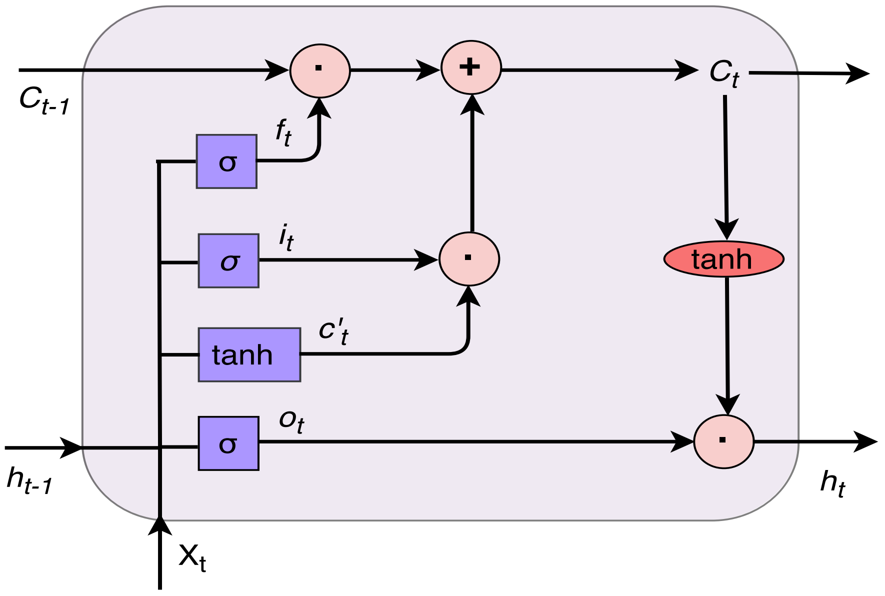


Figure 2: An LSTM cell.

$$\overleftarrow{h}_t = f(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t+1})$$
$$y_t = f(W_{\overrightarrow{h}y}\overrightarrow{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t)$$

Attention mechanism Attention based neural networks are successful in diverse tasks like speech recognition [3] and language translation [4], where the neural network finds it difficult to comprehend the fixed length vector representation of a very long input sequence. We implemented the traditional attention mechanism proposed in [4]. For an input sequence of length n, the importance of each nucleotide position i is accessed by calculating α_i^j for $i = 1, 2, ..., n$ using the formula

$$\alpha_i^j = \frac{exp(e_{ij})}{\sum_{k=1}^{n} exp(e_{ik})}$$

where e_{ij} is the *tanh* activations of the dot products of hidden layer representations (h_i), generated by the BLSTM layer, with the attention layer weights. Based on the importance α_i^j of each hidden state h_i generated for input nucleotide i, a context vector c is generated as

$$c_j = \sum_{i=1}^{n} \alpha_i^j h_i$$

for predicting the output at time step j. At each time step, a new set of attention weights and hidden states are generated. We use the attention weights generated at the n^{th} time step.

2 Experimental Setup

2.1 Efficient implementation of occlusion

Instead of naively iterating through each sequence and then iterating through each window, we prepared the modified sequences for each input sequence beforehand. Thus, for a single input sequence of length n, we obtained the modified n sequences corresponding to each occluded window. We concatenated the n modified

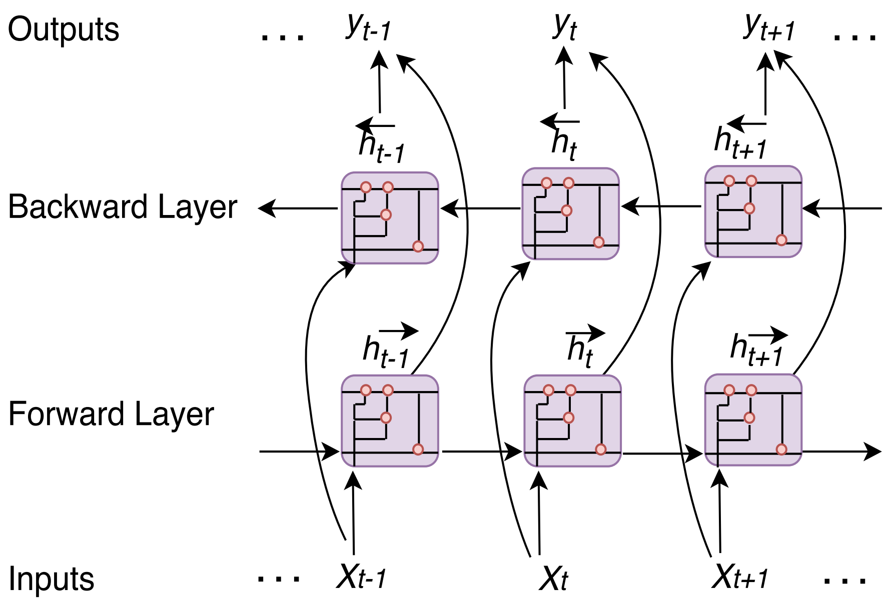


Figure 3: A BLSTM network.

sequences, resulting in a $n \times n$ matrix corresponding to each input sequence. Finally, upon concatenation of all the modified sequences of the B input sequences in a batch, we obtain a $(B * n) \times n$ matrix which is fed in a single batch to the model, to directly predict the probabilities, resulting in a $(B * n) \times 1$ matrix. This is reshaped to get the required $B \times n$ matrix of *deviation values*.

2.2 Data generation

Figure 4 is a pictorial representation of the Type-1 and Type-2 dataset.

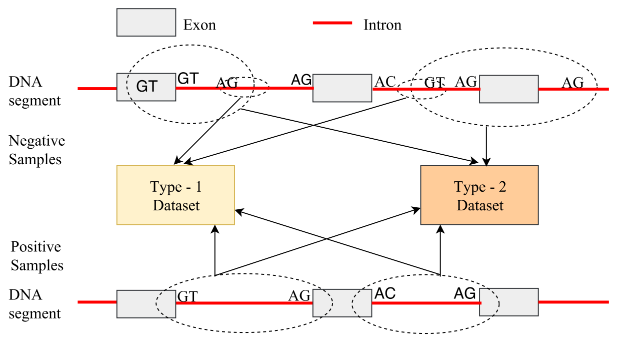


Figure 4: A pictorial representation of the Type-1 and Type-2 dataset.
For simplicity of representation, the entire extracted DNA region for the positive samples and the Type-2 negative samples are displayed using the dashed circles. These samples are truncated to 40 nt upstream and downstream regions before feeding into the learning model.

2.3 Variation in length of flanking region

We vary the length of flanking region in the input sequences to find the optimal region that provides best predictive performance of the model. The flanking region is varied from 10 nt to 50 nt in the upstream and downstream sequences of both the donor and acceptor splice junctions. Results shown are for the Type-2 dataset.

The performance of the model improves with increase in the flanking region, displaying the best performance with 40 nt flanking region. All the analysis produced in the paper are performed with a flanking region of 40 nt. The performance of the model with varying flanking region is shown in Table 1.

Table 1: Performance of the model with variation in flanking region

Flanking region	Performance measures
	$Ac, Pr, Re, F1(\%)$
50 nt	97.97, 98.91, 97.00, 97.95
40 nt	**98.38, 99.38, 97.38, 98.37**
30 nt	97.18, 99.03, 95.29, 97.13
20 nt	96.45, 97.26, 95.60, 96.42
10 nt	95.06, 95.71, 94.35, 95.03

2.4 Variation in model architecture

Table 2 shows the variation in the performance of the model with variation in the number of hidden layers (HL). Results are shown for Type-1 dataset. We show the performance with both LSTM and BLSTM as the units of the hidden layer. We see that the model is invariant to the number of hidden layers as well as the type of units in the hidden layer. All the analysis produced in the paper are performed with a model having one hidden layer of BLSTM units.

Table 2: Performance of the model with variation in architecture

Model	Performance measures
	$Ac, Pr, Re, F1(\%)$
LSTM with 1 HL	98.24, 99.70, 96.77, 98.21
LSTM with 2 HL	98.28, 99.57, 96.97, 98.25
BLSTM with 1 HL	98.24, 99.66, 96.81, 98.21
BLSTM with 2 HL	98.24, 99.66, 96.81, 98.21

2.5 Hyperparameters of the various baselines

- Vanilla LSTM: We consider a batch size of 128, dropout rate as 0.5, recurrent dropout rate as 0.2, and number of epochs as 50.
- LSTM with attention: We consider a batch size of 128, dropout rate as 0.5, recurrent dropout rate as 0.2, and number of epochs as 10.
- SpliceVec-MLP [5]: We consider one hidden layer with 2500 hidden nodes. We used a batch size of 128 and the learning rate of the Adam optimizer as 0.001.
- DeepSplice [6]: We consider a batch size of 160, dropout keep probability as 0.8 and the Adam optimizer learning rate as 0.001.

2.6 Input formation for SpliceVec-MLP

This model extracts the complete intronic sequence, along with the flanking upstream and downstream region, converts it into an N-dimensional feature representation, and then feeds it into the learning model. In Type-1 dataset, the negative splice junctions are generated by extracting a continuous sequence of 164 nt from the middle of each intron. Unlike Type-2 dataset, Type-1 dataset has a fixed length negative input samples.

Hence, to access the performance of SpliceVec-MLP on Type-1 dataset, we have truncated the positive splice junction sequences to 40 nt upstream and downstream region of both the donor and acceptor splice junctions. The 82 nt sequences obtained from both the splice junctions of a junction pair were concatenated to obtain a 164 nt sequence representing a true splice junction.

3 Results

3.1 t-SNE plots of feature representation

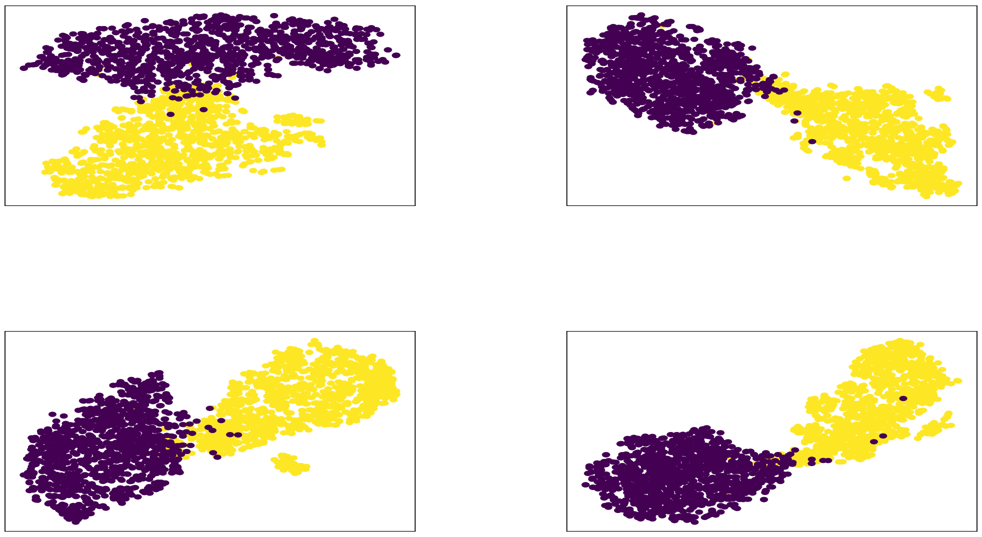
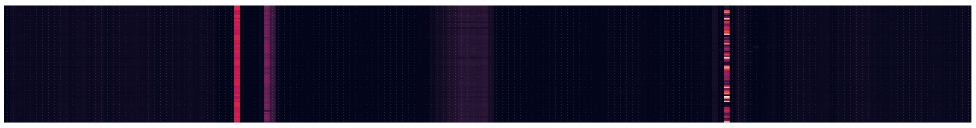


Figure 5: t-SNE plots of 4 different sets of 1000 true and 1000 decoy splice junctions. Points in blue represent true whereas points in yellow represent decoy splice junctions.

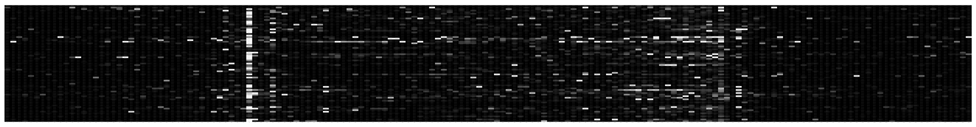
3.2 Heat maps for various visualizations

For each of the visualization techniques, we generate heat maps for each set. The heat maps are generated from the corresponding matrix of *deviation values*, named *deviation matrix*. Further, each set is sampled to obtain a final heat map comprising of random sequences from all the sets. Each sequence in the heat map is 164 nt long. The first 82 nt of a sequence are the upstream and downstream regions of the donor site whereas the next 82 nt are the upstream and downstream regions of the acceptor site of a junction pair. The heat map consists of true test sequences only.

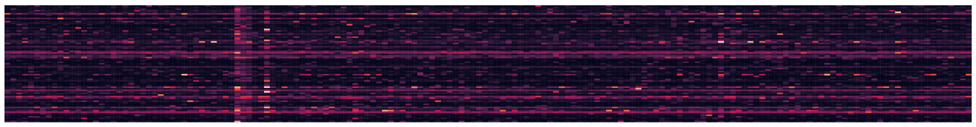
Figure 6 shows the final heat map obtained by sampling all the 15 sets of canonical sequences for all the four types of visualizations employed. The vector positions are colored according to the *deviation values* for each of the 164 sequence indices ranging from -40D to 40D and -40A to 40A.



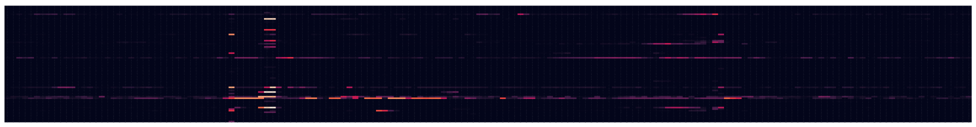
(a)



(b)



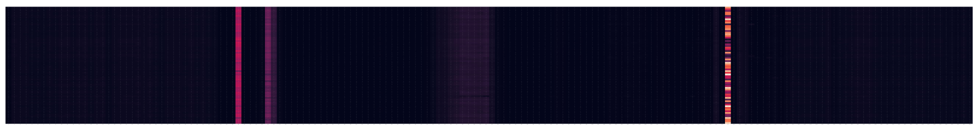
(c)



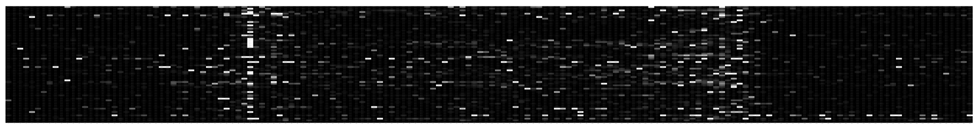
(d)

Figure 6: Heat maps generated by the various visualization techniques for canonical sequences. Each sequence length is 164 nt where the first 82 nt represent the donor site and the next 82 nt represent the acceptor site of a splice junction pair. The heat maps represent: (a) Attention weights (b) Smooth gradients (c) Omission scores (d) 3-mer occlusion weights.

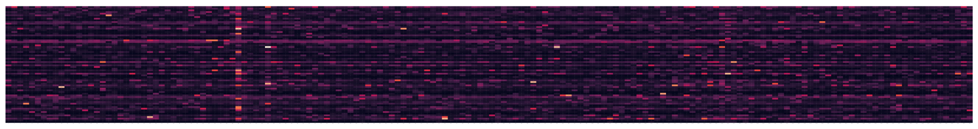
Figure 7 shows the final heat map obtained by sampling all the 7 sets of non-canonical sequences for all the four types of visualizations employed.



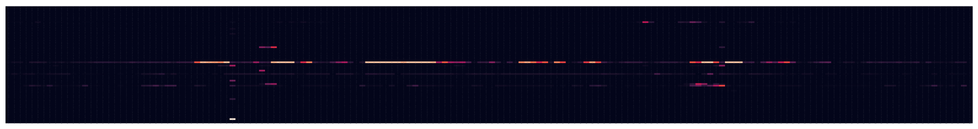
(a)



(b)



(c)



(d)

Figure 7: Heat maps generated by the various visualization techniques for non-canonical sequences. Each sequence length is 164 nt where the first 82 nt represent the donor site and the next 82 nt represent the acceptor site of a splice junction pair. The heat maps represent: (a) Attention weights (b) Smooth gradients (c) Omission scores (d) 3-mer occlusion weights.

3.3 Occurrence-frequency graph of various visualizations

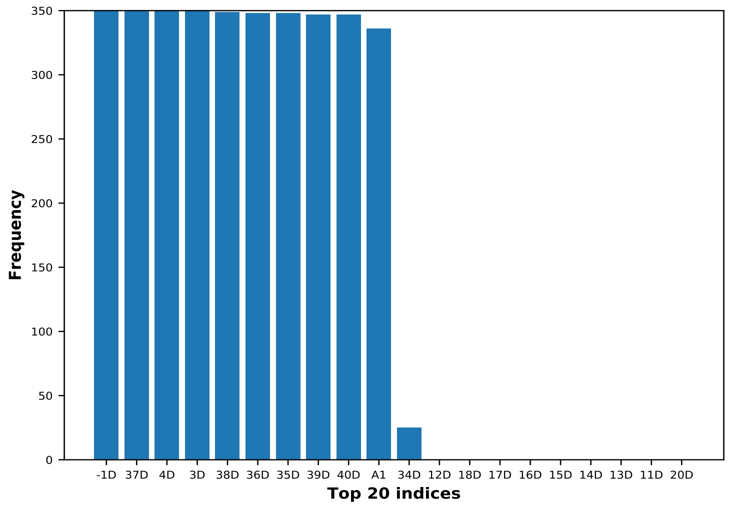


Figure 8: Occurrence-frequency graph based on attention weights of top 20 indices among all the non-canonical true sequences across the 7 sets.

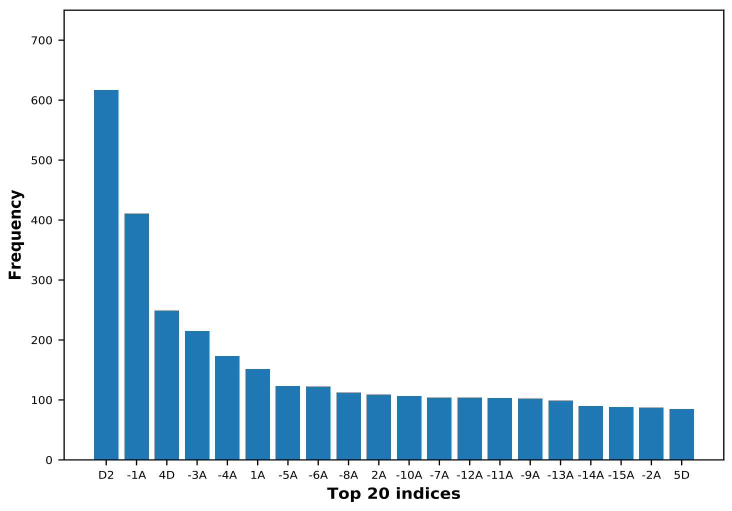


Figure 9: Occurrence-frequency graph based on smooth gradients of top 20 indices among all the canonical true sequences across the 15 sets.

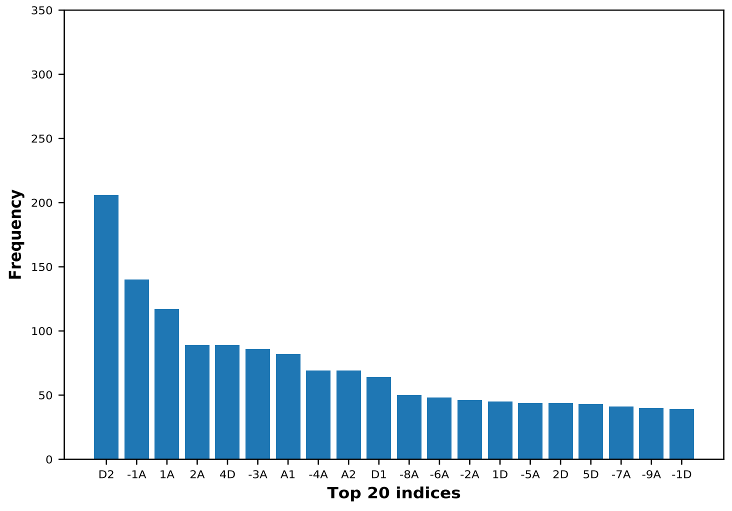


Figure 10: Occurrence-frequency graph based on smooth gradients of top 20 indices among all the non-canonical true sequences across the 7 sets.

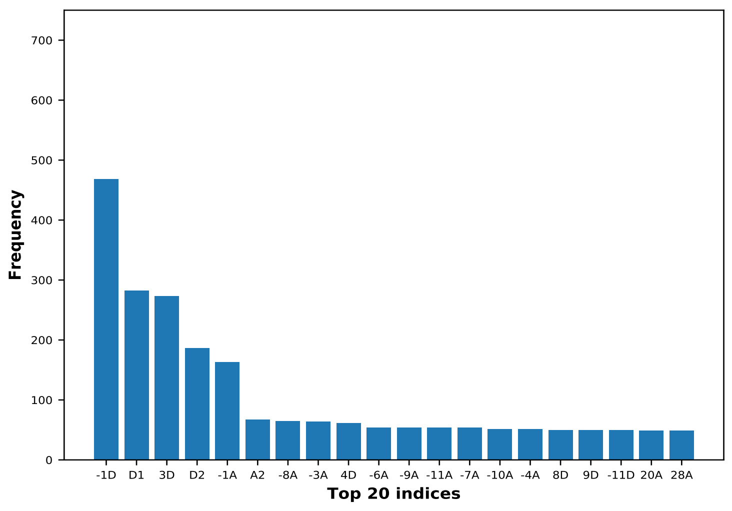


Figure 11: Occurrence-frequency graph based on omission scores of top 20 indices among all the canonical true sequences across the 15 sets.

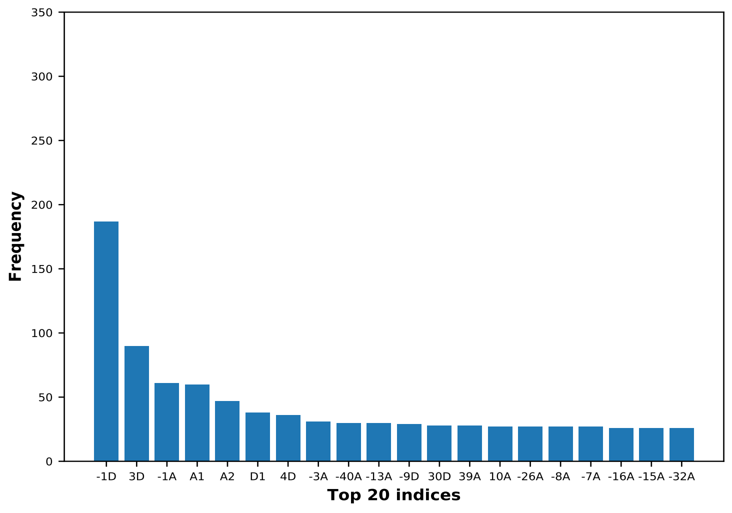


Figure 12: Occurrence-frequency graph based on omission scores of top 20 indices among all the non-canonical true sequences across the 7 sets.

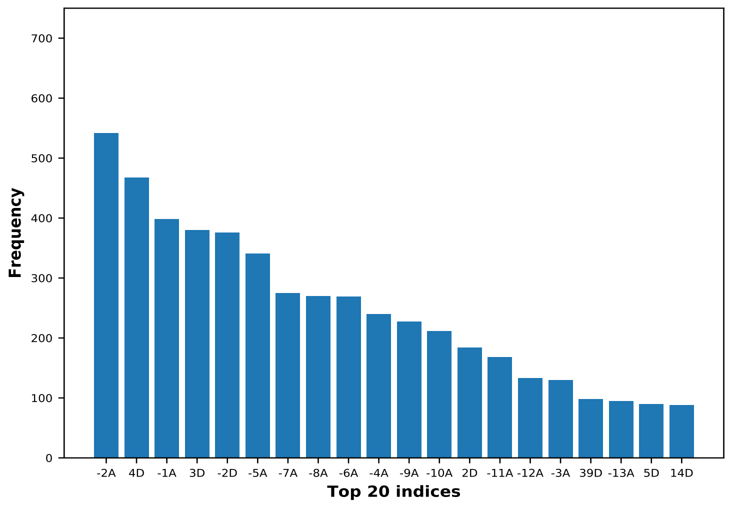


Figure 13: Occurrence-frequency graph based on 3-mer occlusion of top 20 indices among all the canonical true sequences across the 15 sets.

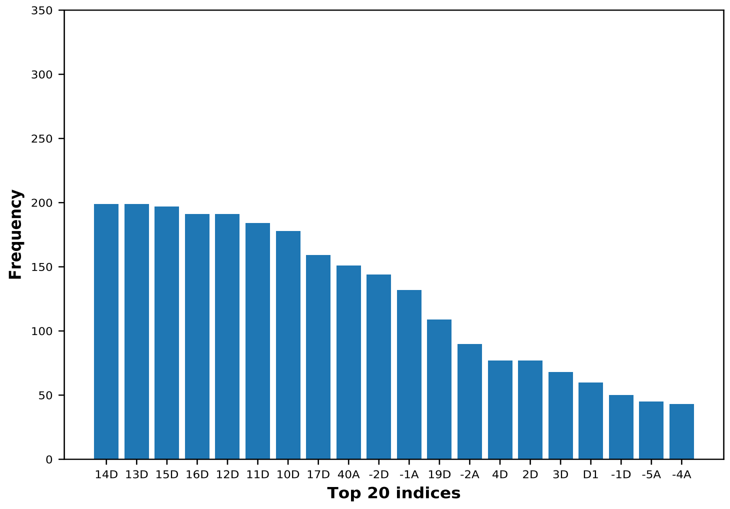


Figure 14: Occurrence-frequency graph based on 3-mer occlusion of top 20 indices among all the non-canonical true sequences across the 7 sets.

Table 3: The indices and sequence patterns obtained from the top 5 sequences for each visualization. A '-' in the sequence pattern signifies a gap of one nucleotide position whereas a '*' signifies a gap of two or more nucleotide positions.

Visualization	Sequence pattern	Indices
Attention weights	C*TC*GGGCGT*C	-2D 2D 3D 34D 35D 36D 37D 38D 39D -1A
	C*CC*GCTGGC*C	-2D 2D 3D 34D 35D 36D 37D 38D 39D -1A
	A*GA*CACAGT*G	-2D 2D 3D 34D 35D 36D 37D 38D 39D -1A
	G*GG*CTTTCT*A	-2D 2D 3D 34D 35D 36D 37D 38D 39D -1A
	G*GG*TCCCGG*G	-2D 2D 3D 34D 35D 36D 37D 38D 39D -1A
Smooth gradients	C*C*C*C*C*C*C*C*C*C	-39D -35D -23D -20D -16D -5D 17D 24D 32D A2
	C*C-C*C*C*C*T*T*T*C	-24D D1 1D 10D 13D 26D 35D -22A -17A 1A
	T*T*T*T*T*T*T*T*T*T	-30D D1 8D 35D -35A -28A -10A -7A -2A A2
	C*C*CC*C*C*C*C*C*C	-9D -4D D1 D2 19D -40A -21A -12A -9A A2
	C*C*C*T*C*C*C*C-C*T	-30D -16D -7D D2 5D 39D -17A -1A A2 3A
Omission scores	A*T*G*C*C*G*C*C*A*A	-23D -11D 2D 19D 40D -34A -15A -7A -1A 4A
	C*C*G*C*GT-A*AG*A	-34D -30D -16D -32A -13A -12A -10A -1A A1 12A
	A*A-C-GA*G*G*G*C*G	-20D -1D D2 2D 3D 19D 26D -31A -19A 13A
	T*GT*A*G*T*G*A*G-A	-31D -22D -21D -3D -31A -19A A1 14A 28A 30A
	C*G*C*C*G*T*A*C*C*C	-40D -16D -9D -6D -2D 2D 23D -27A 3A 36A
3-mer occlusion	GGTA*G-TG-TG*T	-2D -1D D1 D2 -21A -19A -18A -16A -15A -1A
	AG-T*CTTC*AGG	19D 20D 22D 26D 27D 28D 29D -1A A1 A2
	ACCTTATG*GG	7D 8D 9D 10D 11D 12D 13D 14D 12A 13A
	AGTG*A*T*AGT*G	-2D -1D D1 D2 21D 34D -32A -31A -30A -26A
	TTTATT*CTAT	-13A -12A -11A -10A -9A -8A -5A -4A -3A -2A

References

[1] Z.-X. Lu, P. Jiang, Y. Xing, Genetic variation of pre-mrna alternative splicing in human populations, Wiley Interdisciplinary Reviews: RNA 3 (4) (2012) 581–592.

[2] X. H. Zhang, K. A. Heller, I. Hefter, C. S. Leslie, L. A. Chasin, Sequence information for the splicing of human pre-mrna identified by support vector machine classification, Genome Research 13 (12) (2003) 2637–2650.

[3] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, Y. Bengio, Attention-based models for speech recognition, in: Advances in neural information processing systems, 2015, pp. 577–585.

[4] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473.

[5] A. Dutta, T. Dubey, K. K. Singh, A. Anand, Splicevec: distributed feature representations for splice junction prediction, Computational biology and chemistry 74 (2018) 434–441.

[6] Y. Zhang, X. Liu, J. N. MacLeod, J. Liu, Deepsplice: Deep classification of novel splice junctions revealed by rna-seq, in: Bioinformatics and Biomedicine (BIBM), 2016 IEEE International Conference on, IEEE, 2016, pp. 330–333.