Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

Sequence Analysis

SpliceRover: Interpretable Convolutional Neural Networks for Improved Splice Site Prediction

Jasper Zuallaert [1,2,*], Fréderic Godin [2], Mijung Kim [1,2], Arne Soete[3,4], Yvan Saeys [4,5], and Wesley De Neve [1,2]

[1] Center for Biotech Data Science, Ghent University Global Campus, Songdo, Incheon, 305-701, South Korea
[2] IDLab, Ghent University - imec, ELIS, Ghent, 9000, Belgium
[3] Department of Biomedical Molecular Biology, Ghent University, Ghent, Belgium
[4] Data Mining and Modeling for Biomedicine, VIB Inflammation Research Center, Ghent, Belgium
[5] Department of Applied Mathematics, Computer Science and Statistics, Ghent University, Ghent, Belgium

[*] To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: During the last decade, improvements in high-throughput sequencing have generated a wealth of genomic data. Functionally interpreting these sequences and finding the biological signals that are hallmarks of gene function and regulation is currently mostly done using automated genome annotation platforms, which mainly rely on integrated machine learning frameworks to identify different functional sites of interest, including splice sites. Splicing is an essential step in the gene regulation process, and the correct identification of splice sites is a major cornerstone in a genome annotation system.

Results: In this paper, we present SpliceRover, a predictive deep learning approach that outperforms the state-of-the-art in splice site prediction. SpliceRover uses convolutional neural networks (CNNs), which have been shown to obtain cutting edge performance on a wide variety of prediction tasks. We adapted this approach to deal with genomic sequence inputs, and show it consistently outperforms already existing approaches, with relative improvements in prediction effectiveness of up to 80.9% when measured in terms of false discovery rate. However, a major criticism of CNNs concerns their "black box" nature, as mechanisms to obtain insight into their reasoning processes are limited. To facilitate interpretability of the SpliceRover models, we introduce an approach to visualize the biologically relevant information learnt. We show that our visualization approach is able to recover features known to be important for splice site prediction (binding motifs around the splice site, presence of polypyrimidine tracts and branch points), as well as reveal new features (e.g., several types of exclusion patterns near splice sites).

Availability: SpliceRover is available as a web service. The prediction tool and instructions can be found at http://bioit2.irc.ugent.be/splicerover/.

Contact: jasper.zuallaert@ugent.be

Supplementary information: Supplementary materials are available at *Bioinformatics* online.

1 Introduction

The identification of functional sites such as splice sites, start/stop codons, and coding and non-coding regions in the genome constitutes an essential component of current annotation systems (Sterck *et al.*, 2012). These annotation systems highly depend on state-of-the-art machine learning approaches that are used for both identifying the specific functional sites as well as for constructing integrative models that combine different predictions to learn the overall gene structure (Foissac *et al.*, 2008). In particular, splice site prediction models are used to identify donor and acceptor splice sites, which respectively denote the exon-intron and intron-exon junctions. Identifying splice sites is an essential step in elucidating the gene structure, and splice site prediction systems can be used to gain novel insights into alternative splicing events as well as potential splicing defects, e.g., by identifying mutations that would hinder correct splicing (Jian *et al.*, 2013).

The identification of donor splice sites is typically seen as a binary classification problem different from the identification of acceptor splice sites, and where both problems require making a distinction between true splice sites and pseudo splice sites. The corresponding datasets are composed of sequences of a fixed length, representing the regions around canonical splice site dinucleotide patterns (`GT` and `AG` for donors and acceptors, respectively), as they account for almost 99% of known sites (Burset *et al.*, 2001).

In this paper, we introduce SpliceRover, an approach that leverages convolutional neural networks (CNNs) for splice site prediction. CNNs are highly suitable for pattern recognition tasks, and they have been successfully applied in many state-of-the-art solutions for image processing problems (Krizhevsky *et al.*, 2012) and natural language processing tasks (Kim, 2014). However, deep neural networks often behave as black box models, yielding limited interpretability of the decision-making processes carried out by these models.

This work presents the following contributions:

- We outline an effective CNN-based architecture for genomic data analysis, outperforming the state-of-the-art for the use case of splice site prediction.
- We conduct a systematic and comparative evaluation of our end-to-end learning approach using four different datasets, consisting of arabidopsis thaliana or human DNA samples, targeting both donors and acceptor splice sites. For the sake of reproducibility, we added all experimental details as supplementary materials.
- We provide detailed insights into the inner workings of our models, visualizing and interpreting a variety of biologically relevant features they automatically learn without any prior knowledge. Our visualizations produce qualitatively coherent results, verifiable against known hypotheses that have been put forward by human experts.
- We provide a publicly available prediction tool as a web service, for both donor and acceptor splice site prediction, aiming at both the human and the arabidopsis genome.

2 Related Work

We can divide existing methods for splice site prediction into three major categories: (1) probabilistic methods; (2) alignment-based methods; and (3) machine learning methods.

Earlier approaches utilize probabilistic models to perform predictions, for instance taking advantage of Markov models (Pertea *et al.*, 2001). Alignment-based approaches use reads from RNAseq for measuring gene expression levels and splice site prediction (Trapnell *et al.*, 2009; Wang *et al.*, 2010). Recently, machine learning has become increasingly popular, as the steep increase in cheap computational power makes the use of computationally complex models feasible. Moreover, given that sequencing techniques are becoming cheaper and more effective, more sequencing data and annotations are available, which means that complex models can take advantage of more data to train on.

A well-known machine learning technique consists of the use of support vector machines (SVM). While being a popular technique, an SVM-based approach often still makes use of manually defined features (Degroeve *et al.*, 2005; Bari *et al.*, 2014), though string kernels are available for pattern analysis (Sonnenburg *et al.*, 2007). Other machine learning techniques allow for automatic feature extraction, and these techniques have been used for splice site prediction as well, including an approach based on restricted Boltzmann machines (Lee and Yoon, 2015). CNNs also facilitate automatic feature extraction. With respect to splice sites, CNNs have been used for combined donor/acceptor prediction (Zhang *et al.*, 2016) and branch point prediction (Dean *et al.*, 2016; Paggi and Bejerano, 2017), though model interpretability is provided only to a limited degree. In addition, they have been deployed in a wide range of genomic analysis use cases, such as predicting DNA-protein binding (Zeng *et al.*, 2016), sequence classification (Nguyen *et al.*, 2016), alternative splicing pattern prediction (Leung *et al.*, 2014), and others (Alipanahi *et al.*, 2015; Sønderby *et al.*, 2015; Quang and Xie, 2016).

To address the lack of understanding of the decision-making processes executed by neural networks, multiple studies have been conducted on visualizing learnt features. Different approaches utilize occlusion masks or mutation maps to measure sensitivity in the input (Zeiler and Fergus, 2014; Alipanahi *et al.*, 2015). Zeiler and Fergus (2014) also define transposed convolutions to create so-called *deconvnets*, which allow propagating a specific activation back to the input layer, yielding visualizations of specific neurons or convolutional filters. Other approaches calculate the gradient to measure which input signals have the greatest impact on a specific prediction (Bach *et al.*, 2015). Shrikumar *et al.* (2017) present DeepLIFT, which addresses the shortcomings of the aforementioned visualization approaches, calculating contribution scores by comparing the activation of each neuron to a reference activation.

However, while being intuitive in the visual domain, it is not straightforward to interpret the calculated scores in genomic data to determine the decisive prediction factors. Therefore, the research effort presented in this paper utilizes DeepLIFT to visualize the reasoning of our neural network models in an interpretable way.

3 Proposed Approach

In this section, we outline our approach for predicting splice sites. We first give an introduction to CNNs, followed by an overview of the network architecture designed. Finally, we discuss the visualization algorithm used.

Convolutional neural networks

Artificial feed-forward neural networks (NNs) are computer models designed for the representation of high-level abstractions in data, largely based on how the human brain works (LeCun *et al.*, 2015). These models consist of different layers, each holding a number of neurons. Each neuron consists of a number of parameters (weights). Typically, a network consists of an input layer, a number of hidden layers, and an output layer. Input data are propagated through the network, yielding activations (intermediary results) for each hidden layer and resulting in a final prediction at the last layer. To create a more complex model that enables the representation of non-linear functions, a non-linearity is applied to the activations of each layer (e.g., a rectified linear unit). Typically, in classification problems, the last layer uses the softmax function in its calculations, generating a probability for each class.

NNs learn from annotated training data by adjusting the weights based on a cost function. This cost function represents the difference between the predictions of the network and the annotated labels. However, a hidden layer of a NN has individual, independent weights for every position in the input it receives. This implies that it is not possible to learn to look for a particular pattern over a whole sequence. For that purpose, CNNs were introduced. These are NNs with one or more convolutional layers. Each of these layers has a number of filters, sliding over the sequence and detecting patterns. Here, weights are stored within a filter to be shared over different positions.

To recognize a particular pattern on different positions, we also need to reduce the dimensionality throughout our network. By doing so, an activation in the final layers of a network holds information over a range of

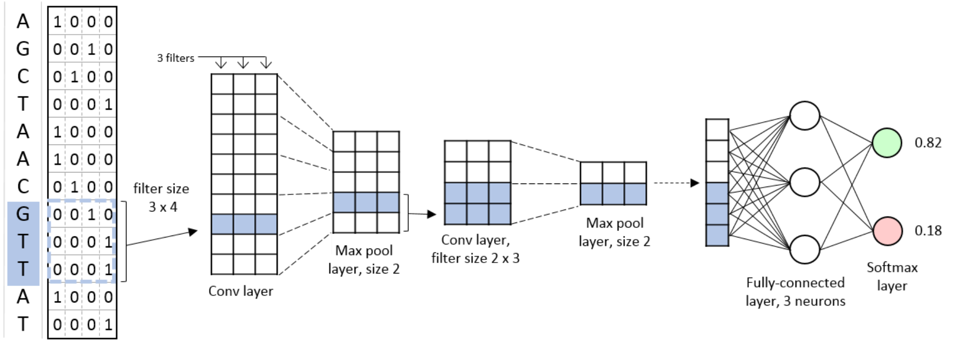


Fig. 1: Our proposed network architecture. Input vectors are fed to a first convolutional layer, with filter size $k \times 4$ (with k denoting the length of the input sequence), After the first convolutional layer, a max-pooling layer is applied, followed by a number of alternating convolutional and max-pooling layers. After the last max-pooling, all filter outputs are fed to a fully-connected layer. If necessary, extra fully-connected layers can be added. Finally, the last fully-connected layer is connected with a softmax classifier.

positions. To reduce the dimensionality of an input sequence, subsampling can be applied by combining neighboring activations into a single value. An example is max-pooling, where the maximum activation over a number of positions is kept. Like this, dimensionality is reduced, whilst keeping track of the most significant activations.

Network architecture

To detect discriminatory patterns in DNA sequences that lead to correct predictions, we constructed a CNN. An illustration of our network architecture is given in Fig. 1. Given the use of filters, sliding over the input layer, the need for manually defined positional and compositional features is eliminated. The network automatically learns which features are critical.

Training is conducted on a set of raw DNA sequences, which we convert to numerical vectors using a one-hot encoding (e.g., `A` corresponds to the vector $[1, 0, 0, 0]$). Next, a number of alternating convolutional, dropout, and max-pooling layers are applied, followed by one or more fully connected layers, to conclude with a softmax classifier. The classifier outputs a number between 0 and 1 for both the positive and negative classification, indicating the probability it assigns to those predictions.

In Section 4, we provide the exact parameters of our topology.

Visualization and interpretation

To achieve the visualization of the features learnt by our network, we make use of the DeepLIFT algorithm, as recently proposed by Shrikumar *et al.* (2017). Given a particular prediction, DeepLIFT allows computing an individual contribution score for each nucleotide, based on the difference of the output from some 'reference' output in terms of differences of the inputs from their 'reference' inputs, by backpropagating that difference through the network.

Our approach consists of three steps:

1. Computation of contribution scores
2. Normalization of contribution scores into weighted contribution scores
3. Aggregation of weighted contribution scores so to be able to identify regions/structures of interest

Firstly, for every nucleotide of an input sequence, a contribution score is calculated by making use of the Rescale Rule of the DeepLIFT algorithm. As a reference input, the strategy as proposed by the authors of DeepLIFT is used, which uses the expected frequency of every nucleotide at each position. To that end, we calculate the averages of the one-hot encodings at every position over the negative training set. For more details on the usage of DeepLIFT, we would like to refer the interested reader to Shrikumar *et al.* (2017).

The obtained contribution scores express the importance of the corresponding nucleotides for the final prediction. This results in contribution scores for all nucleotides of each input sequence in a given dataset. However, these scores do not always act on the same scale for different models. Therefore, we normalize the scores for all sequences in a given dataset.

The formula for normalizing to weighted contribution scores is as follows:

$$wcs_{ij} = 100 * m * \frac{cs_{ij}}{\sum_{p=1}^{m} \sum_{q=1}^{n} |cs_{pq}|}, \tag{1}$$

with cs_{ij} denoting the contribution score for the nucleotide in sequence i at position j, wcs_{ij} denoting the weighted contribution score at that position, m denoting the number of samples in the dataset, and n denoting the sequence length. That way, each weighted contribution score indicates the importance of the corresponding nucleotide towards the prediction made, and where this importance is expressed as a percentage.

The absolute values of the percentages of one sequence add up to 100% if the prediction for that particular sequence by the network is made with an average confidence. If it has more discriminative features, resulting in a higher confidence, then the sum will be higher than 100%, and analogously, in case of a lower confidence, the sum will be lower than 100%. Like that, the scores for different datasets can be normalized to the same scale, whilst also providing an interpretation of the resulting numbers.

For example, given a dataset with sequences `ACTG` and `CGTG` with respective contribution scores (0.1, −0.2, 0.3, −0.05) and (0.5, 0.6, 0.15, −0.01), this would result in (10, −20, 30, −5) and (50, 60, 15, −10) after normalization. As can be seen, the absolute values of the weighted contribution scores of the second sequence add up to a total of 135, indicating that the features in this sequence result in a prediction with higher confidence.

Finally, we produce visualizations by aggregating the weighted contribution scores of all sequences in a dataset in four different ways:

1. Average contribution score per position: At each nucleotide position, we calculate the average contribution score for all nucleotides occurring at that position, regardless of their type.
2. Average/median contribution score per position per nucleotide: At each nucleotide position, we calculate the average contribution score

per nucleotide. This means we achieve an average/median score at that position per type of nucleotide.

3. Most important patterns in a specific region: We specify a pattern length k and a region [a,b]. All k-mers occurring in that region are extracted from the dataset, and the contribution score for each occurrence is calculated, by summing up the individual contribution scores. Next, for each possible pattern (4^k possibilities), we calculate the average/median contribution score in the the extracted occurrences. Finally, we select the patterns with the most positive and most negative average/median contribution scores.
4. Average contribution score of a specific pattern in different regions in the sequence: We specify a pattern, of which we extract the occurrences at every possible starting position, whilst also calculating their corresponding contribution scores. Next, at each starting position, we calculate the average/median contribution score of the extracted occurrences. Finally, as an optional extension, we also group together multiple starting positions into regions. This is further clarified in the Supplementary Materials (Section 2.5).

4 Experimental Setup

To evaluate the effectiveness of the proposed predictive models, we compare our approach to a number of state-of-the-art techniques. In this section, we first give an overview of the datasets and metrics used. For detailed information about the composition of each dataset, we refer the interested reader to the respective papers. The size and class distribution of each dataset can be found in Table 1. Next, we give an overview of the parameters of our network architecture and the settings used by our training procedure. Finally, we discuss our setup for visualizing the decisive features in our splice site prediction models.

Data, metrics, and benchmark methods

Degroeve *et al.* (2005)

The authors define SpliceMachine, an approach that uses linear SVMs with manually defined positional and compositional features. Tests are conducted on an arabidopsis thaliana donor and acceptor splice site dataset. Ten-fold crossvalidation is applied. Effectiveness is measured in terms of the precision (Pr) for a sensitivity (Se) or recall of 0.95. We will refer to this metric as *Pr.95*. To quantify relative improvements, we take into account the false discovery rate (FDR) for a sensitivity of 0.95, which is equal to 1-*Pr.95*. We refer to this metric as *FDR.95*.

$$Se = \frac{TP}{TP+FN} \qquad Pr = \frac{TP}{TP+FP} \qquad FDR = \frac{FP}{TP+FP},$$

where TP denotes true positives, FP false positives, FN false negatives and TN true negatives.

Sonnenburg *et al.* (2007)

The authors use SVMs for splice site recognition by making use of a weighted degree kernel with shifts complemented by six spectrum kernels. Tests are conducted on the Genome Wide Human (GWH) dataset, containing sequences of 140 nucleotides. 5-fold crossvalidation is applied. In each of the five validations, the negative samples in the training set are subsampled by a factor of 5. However, the pos:neg ratios of the validation and test sets are kept at their original values. Effectiveness is measured in terms of area under the precision-recall curve (auPRC).

Bari *et al.* (2014)

The authors use a self-defined encoding scheme with an SVM to predict splice sites in the NN269 dataset. In the acceptor set, sequences have a length of 90 nucleotides (68 nucleotides preceding and 20 nucleotides succeeding the splice site). In the donor set, sequences have a length of only 15 nucleotides (7 in front of and 6 behind the splice site). The data are divided into a set for training and a set for testing. Effectiveness is measured in terms of sensitivity, specificity (Sp), accuracy (Acc), and area under the receiver operating characteristic curve (auROC).

$$Sp = \frac{TN}{TN+FP} \qquad Acc = \frac{TN+TP}{TN+TP+FP+FN}$$

Lee and Yoon (2015)

The authors tackle the problem of splice site prediction using deep belief networks (DBNs), consisting of stacked restricted Boltzmann machines. The encoding strategy is a unigram one-hot encoding. Tests are conducted on the GWH dataset. For training and evaluation, each chromosome is treated separately. In particular, for each chromosome, the negative samples are randomly subsampled to achieve a pos:neg ratio of 1:5. Next, 10-fold crossvalidation is conducted, leading to a total of 240 tests. For each test, the F1 score is calculated.

$$F1 = \frac{2*Se*Pr}{Se+Pr} = \frac{2*TN}{2*TN+FN+FP}$$

The experiments performed are equivalent to the use of 24 separate datasets of different sizes. There is an important difference in the size of these datasets. Where the average size of the datasets used is equal to 19 965 samples, the smallest set consists of 1458 samples, and the biggest set of 47 736 samples.

Model parameters and training procedure

Our proposed architecture consists of five convolutional layers. The first layer uses 70 filters of size 9×4 over the input. The next convolutional layers have 100, 100, 200, and 250 filters for each layer, respectively, all of size 7×1. The third, fourth, and fifth convolutional layer are followed by subsampling, taking the form of max-pooling layers having a size of 3×1, 4×1, and 4×1, respectively. The last convolutional layer is

authors	dataset	origin	# pos	# neg	pos:neg	seq length	metrics
Degroeve *et al.* (2005)	arabidopsis donors	plant	9208	263 507	1 : 28.6	402	*Pr.95*
	arabidopsis acceptors		9310	227 225	1 : 24.4	402	*Pr.95*
Sonnenburg *et al.* (2007)	GWH donors	human	160 600	76 335 126	1 : 475.3	141	auPRC
	GWH acceptors		158 217	54 469 622	1 : 344.3	141	auPRC
Bari *et al.* (2014)	NN269 donors	human	1324	4922	1 : 3.7	15	sensitivity, specificity, accuracy, auROC
	NN269 acceptors		1324	5553	1 : 4.2	90	sensitivity, specificity, accuracy, auROC
Lee and Yoon (2015)	GWH donors	human	80 515	402 575	1 : 5	398	F1
	GWH acceptors		79 250	396 250	1 : 5	398	F1

Table 1. Characteristics of the datasets used in our experiments.

followed by a fully-connected layer with 512 neurons. We end by adding a softmax layer with an output for the positive class and an output for the negative class. Furthermore, all convolutional and fully-connected layers are succeeded by a rectified linear unit (ReLU). During training, a dropout layer ($p = 0.2$) is also added behind each of the first two convolutional layers, each max-pooling layer, and the fully-connected layer.

For training, we used the categorical cross-entropy cost function. In addition, training was done using stochastic gradient descent with nesterov momentum, starting at a learning rate of 0.05. Every five epochs, we divided the learning rate by two, until we reached 50 epochs. Finally, we selected the model that yielded the best validation result.

For input sequences with a length of 205 or less, we scaled down the pooling sizes of our subsampling layers, as the sequence length after the last pooling layer would otherwise drop below two, which would too greatly decrease the expressive capacity of the network. Additionally, as the samples in the NN269 donors set only have a sequence length of 15, we also downscaled the number of layers and the filter sizes. The exact details can be found in the Supplementary Materials (Section 1).

Visualization

For visualizing the features learnt, we ran tests on four distinct datasets: arabidopsis donor, arabidopsis acceptor, GWH donor, and GWH acceptor. Given that we want to compare the results obtained for the aforementioned datasets, we performed random subsampling on each dataset in order to achieve an equal pos:neg ratio of $1:20$. Besides that, we only made use of samples representing canonical splice sites.

We divided each of the datasets into a set for training, validation, and testing, using a 3/1/1 distribution. Each dataset yields an individual model, by first training on the training set and then selecting the optimal model using the validation set. Finally, we use the sequences in the test set for visualization purposes.

Test setup

All tests were conducted on a system with 64 GB of RAM, a 1 TB SSD, and 4 Titan X GPUs, each with 12 GB of memory. We made use of the Theano (v0.9), Lasagne (v0.2), and Keras (v1.1.1) Python packages for training and testing, and of the DeepLIFT (v0.4.0) software package (Shrikumar *et al.*, 2017) for feature visualization.

5 Results and Discussion

Comparison with other approaches

We compare our approach with the state-of-the-art on all of the datasets listed in Section 4. The obtained results are shown in Table 2.

As can be seen in Table 2a, we significantly outperform the SpliceMachine models with ours. On the donors data set, we decrease the *FDR.95* from 0.20 to 0.044, which is a relative decrease of 78.0%. On the acceptors dataset, we are able to decrease the *FDR.95* from 0.32 to 0.061, yielding a relative decrease of 80.9%. It is clear that our models are able to automatically learn decisive features, substantially outperforming the aforementioned linear SVMs using manually defined features.

Results for the Sonnenburg *et al.* benchmark are given in Table 2b. For the donors set, we are able to increase the auPRC from 0.5469 to 0.6194, an increase of 13.2%. When testing on the acceptor set, we increase the auPRC from 0.5412 to 0.5960, yielding a relative improvement of 10.1%.

As can be seen in Table 2c, the difference in effectiveness between our model and the Bari *et al.* benchmark is negligible for the donors set. We observe an auROC of 0.9830 and 0.9829 for the benchmark with polynomial kernel and SpliceRover, respectively, resulting in a relative increase in (1-auROC) of 0.6%.

The similarity in effectiveness in terms of auROC is related to the length of the sequences. Indeed, sequences in the donors set contain less information than sequences in the other datasets, as the former only consist of 15 nucleotides. The NN269 acceptors dataset contains 90 nucleotides per sequence. Therefore, we expect to see a higher effectiveness on this set for our model. Here, we obtain an increase in auROC from 0.9791 to 0.9899, resulting in a relative decrease of 51.7% in terms of (1-auROC). Further confirmation of our hypothesis can be found in the Supplementary Materials (Section 1.4), where we compare both approaches over datasets with an increasing sequence length.

In Table 2d, we enlist the medians of the results of the 10-fold crossvalidation on 24 chromosomes, in terms of F1 score, both for our models and the Lee *et al.* benchmark. The median for donor splice sites increases from 0.816 to 0.907, implying a relative decrease in error in terms of F1 score (1-F1) of 49.5%. For acceptor splice sites, we see an increase from 0.753 to 0.873, resulting in a relative error reduction of 48.6%.

Visualization and interpretation

In this section, we present our visualizations based on four models trained independently on four distinct datasets. The prediction effectiveness of these models is presented in Table 3.

To interpret the decisions made by the networks, we evaluate five hypotheses. Extra visualizations can also be found in the Supplementary Materials (Section 3).

1. **The nucleotides in the proximity of a splice site have the highest impact on the prediction outcome. The regions around these nucleotides are also more influential than the regions at the edges of the input sequence.**

Fig. 2 visualizes the nucleotide importance per position for the human donors dataset and the human acceptors dataset. The nucleotides around the splice site have the highest average influence on the result. The wider regions around the splice site are generally also more important than the edges. Particularly, in front of an acceptor site, this difference in contribution is noticeable, as it is caused by the presence of the polypyrimidine tract (PPT, see Hypothesis 5). The importance per position graphs for the arabidopsis datasets confirmed the observations made for the human datasets.

Additionally, the contribution score of the nucleotides `GT` or `AG` that embody the splice site, will always be zero, as all the input samples have the same nucleotides at that position.

2. **At the center of the sequence, the networks look for a splice site pattern that resembles well-known splice site motifs.**

In Fig. 3, we compare the average weighted contribution scores around the splice site to well-known splice site motifs (Stephens and Schneider, 1992). Specifically, we study the region $[-3, 5]$ of the donor sites, and the region $[-10, 1]$ of the acceptor sites. The bar charts represent the average scores per nucleotide. We found that on each position studied but one (for donors at position -3), the most frequent nucleotide (as depicted in the sequence motif) is also always the nucleotide with the highest average score at that position in our calculations.

In general, the proportions between the weighted contribution scores of the different nucleotides are also similar to the proportions between the nucleotide frequencies. One exception can be found at donors position 2, where we have a positive average score for `A` but not for `G`, even though the sequence motif suggests an almost equal distribution of `A`s and `G`s at that position. If we look at that position in the dataset used for the visualization,

	(a)		(b)		(d)	
		Pr.95		auPRC		med F1 score
donors	SpliceMachine	0.80	Sonnenburg *et al.*	0.5469	Lee *et al.*	0.816
	SpliceRover	**0.956**	SpliceRover	**0.6194**	SpliceRover	**0.907**
acceptors	SpliceMachine	0.68	Sonnenburg *et al.*	0.5412	Lee *et al.*	0.753
	SpliceRover	**0.939**	SpliceRover	**0.5960**	SpliceRover	**0.873**

(c)

		Se	Sp	Acc	auROC
donors	Bari *et al.* (Poly)	0.8798	**0.9719**	0.9525	**0.9830**
	Bari *et al.* (RBF)	0.8894	0.9693	0.9525	0.9824
	SpliceRover	**0.9011**	0.9674	**0.9535**	0.9829
acceptors	Bari *et al.* (Poly)	0.7740	0.8716	0.9339	0.9790
	Bari *et al.* (RBF)	0.7930	0.8728	0.9358	0.9791
	SpliceRover	**0.9077**	**0.9739**	**0.9612**	**0.9899**

Table 2. Results obtained for the various datasets, compared to the benchmark results. Comparisons are made with (a) SpliceMachine on the arabidopsis dataset, (b) Sonnenburg *et al.* on the GWH dataset, (c) Bari *et al.* on the NN269 dataset, listing the results for a polynomial kernel and an RBF kernel, and (d) Lee *et al.* on the GWH dataset.

dataset	auPRC
arabidopsis donors	0.9861
arabidopsis acceptors	0.9611
GWH donors (human)	0.9398
GWH acceptors (human)	0.9550

Table 3. Evaluation results obtained for the four models on the test set to be used for visualization.

we see that 58% of all positives contain an A at that position, and 35% contain a G. Additionally, we see that 12% of the sequences with an A at that position are labeled positive, yet only 5% of the sequences with a G are labeled positive. These observations explain the difference between both nucleotides in our visualization.

Additionally, the acceptor model clearly learns the characteristics of the PPT preceding each acceptor site. In this region, they achieve a positive average weighted contribution score for T and C content. This is confirmed by the positive samples in our data, where we have a C+T rate of 76% for humans in region $[-18, -5]$.

3. **The networks automatically learn to distinguish exon content from intron content. For a positive classification, the content at the left and right side of the splice site under investigation should conform to the characteristics of introns or exons.**

To evaluate this hypothesis, we plot the median weighted contribution scores per position, rather than the averages, to rule out the influence of outliers that are part of a bigger pattern. Amit *et al.* (2012) suggest an increase of G+C content in exons as compared to introns in arabidopsis.

Fig. 4 confirms that our arabidopsis donor model indeed learns the aforementioned property. G+C content yields positive scores in front of the splice site (exon), and negative scores behind the splice site (intron). The opposite is true for T+A content.

The opposite effect occurs for our arabidopsis acceptor model. G+C content yields positive scores behind the splice site (exon) and negative scores in front of the splice site (intron). For Ts, this effect particularly occurs in front of the splice site, due to the abundance of Ts in the PPT.

4. **The networks are highly sensitive to other potential splice sites occurring in the remainder of the sequence, decreasing or increasing the possibility that the currently investigated splice site is indeed a true splice site.**

Apart from single nucleotides, our networks also take into account longer patterns when predicting splice sites.

We make a distinction between patterns preceding the potential splice site and patterns succeeding the splice site. Fig. 5 shows patterns with a length of six nucleotides, yielding the highest and lowest average weighted contribution scores for the arabidopsis donors model. A pattern size of six was chosen to maintain a significant number of occurrences per pattern in our dataset. The most positive patterns preceding the splice site again justify Hypothesis 3, as the fifteen highest scoring patterns in front of the site consist for 70% of G+C content, and the fifteen highest scoring patterns behind the site for 81% of T+A content.

More interestingly, the fifteen most negative average scores are of a much greater magnitude than the positive ones. For example, the GGTAAG pattern contributes negatively for 41.2% on average to the final prediction when preceding the splice site, whereas the CCAGGA pattern only contributes for 2.3%. We conclude that the model is highly sensitive to the negative patterns shown in this plot. Furthermore, when we take a closer look at the patterns, we can see that they actually represent another donor splice site.

Fig. 6a gives an overview of how the CAG**GT**AAG pattern contributes over different positions, for arabidopsis donors. The pattern scores negatively over the whole sequence. This is as expected, because the sequences are too short to contain both a full exon and a full intron in the first or the last 200 nucleotides, which would yield the possibility of the presence of a second true donor site. As a result, as soon as the model encounters the CAG**GT**AAG splice site pattern in another place, it will conclude that the site in question is much more likely to be a true donor site, resulting in a negative prediction for the splice site under investigation.

5. **Additional biological features are recognized by the networks, such as the polypyrimidine tract (PPT) located upstream of an acceptor splice site, and the presence of a branch point.**

In Hypothesis 2 and Hypothesis 3, we already indicated that our network learns several characteristics of the PPT in front of an acceptor site. In addition, we know that a branch site should be located upstream of the PPT. This branch site is located in the region $[-36, -23]$ for humans in approximately 83% of the cases (Gao *et al.*, 2008). A typical branch site is represented by the nucleotide pattern CTNA (Iwata and Gotoh, 2011).

In Fig. 6b, we investigate the influence of the typical branch site pattern CTNA (Iwata and Gotoh, 2011). We observe a clear peak in the aforementioned region, which is marked in the graph. Furthermore, we observe that the branch site pattern scores negatively when it occurs in the proximity of the splice site under investigation. This confirms that the network looks for a branch point in the correct region.

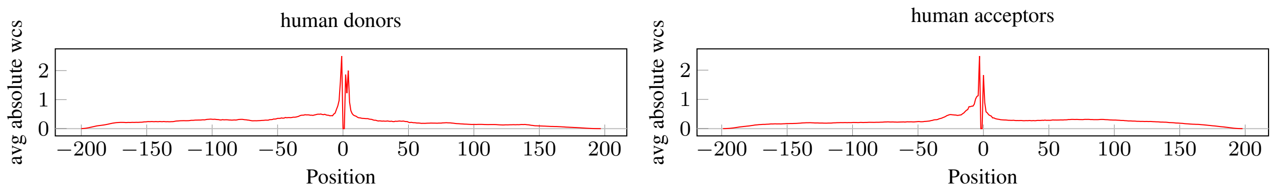
An extra characteristic upstream of acceptor sites is the presence of an AG exclusion zone (AGEZ), as suggested by Gooding *et al.* (2006). For humans, the AGEZ spans the region between the twelve nucleotides before the branch point, up to the acceptor site. Thus, when investigating the region $[-23, 2]$ in Fig. 6c, a sharp drop in median weighted contribution score occurs for the pattern AG.

6 Conclusions

In this paper, we introduced SpliceRover, a CNN-based end-to-end learning approach for splice site detection, facilitating automatic feature extraction and classification of genomic sequences as true or pseudo splice sites. Specifically, we designed and implemented a multi-layered convolutional neural network architecture, taking genomic sequences as one-hot vectors as input, and generating probabilities for a positive and negative classification as output. When compared to four state-of-the-art splice site prediction techniques, our trained models performed better.

Furthermore, we provided an in-depth analysis of the choices made by our network models, demonstrating that our models learn biologically relevant properties of splice sites, exons, and introns. In particular, we presented five hypotheses, which we were all able to prove, mainly through interpreting visualizations. In that context, we could observe that most of the decision-making is happening in the proximity of the splice site itself, where we found that our models learnt the well-known splice site motifs. Also, the arabidopsis models were able to make a distinction between exons and introns by comparing the G+C and A+T content of the different regions in the input sequence. Besides that, we could observe that our models are highly sensitive to other splice site patterns present in the sequence, greatly lowering the odds of the currently investigated splice site to be seen as a real splice site when this situation occurs. Finally, our models also learnt the properties and the position of the polypyrimidine tract and the branch point preceding an acceptor splice site.

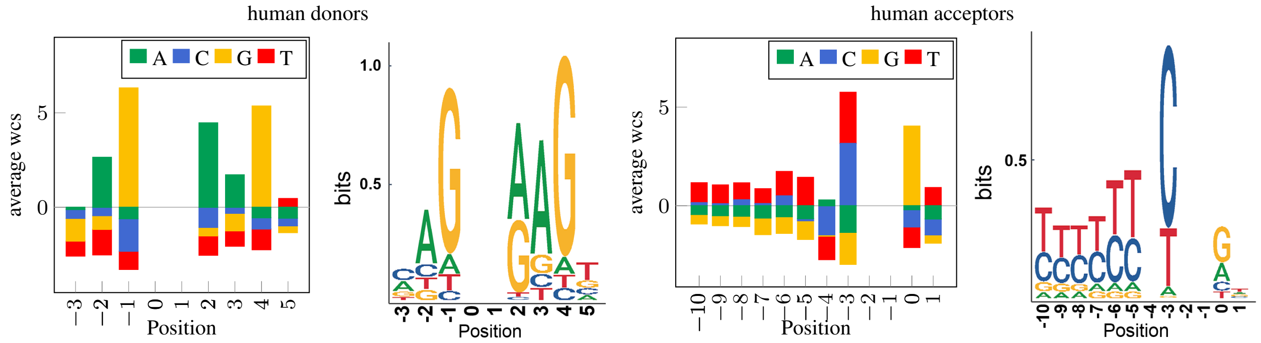
All visualizations were obtained for models that did not possess any biological knowledge prior to training on the datasets used. As a result, we can state that our visualizations help in interpreting the underlying decision-making processes of the CNN-based predictive models, which typically have a black box nature. At the same time, we believe that our visualizations open up a door towards understanding biological questions of which we currently have little knowledge, by granting the possibility of training a network on raw genomic data and evaluating which features that network is sensitive to, a direction we plan to pursue in future research.



human donors

human acceptors

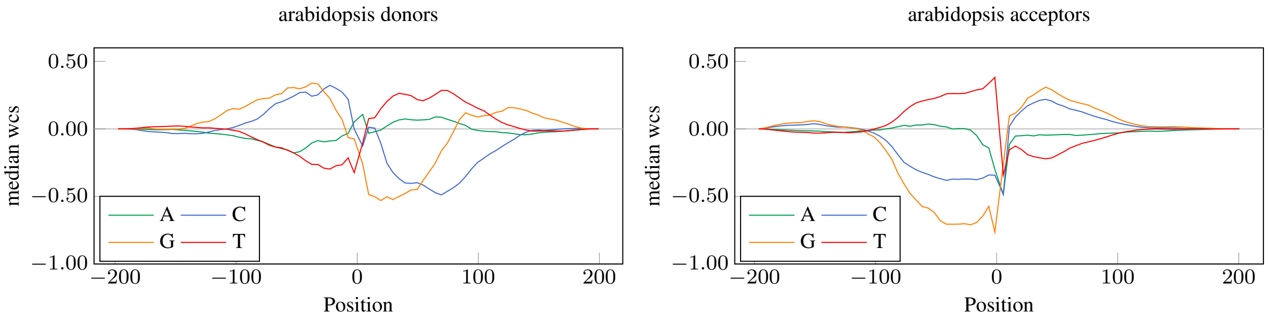
Fig. 2: Importance per position. The average absolute weighted contribution score for each nucleotide position is shown for our human donors and acceptors models.



human donors

human acceptors

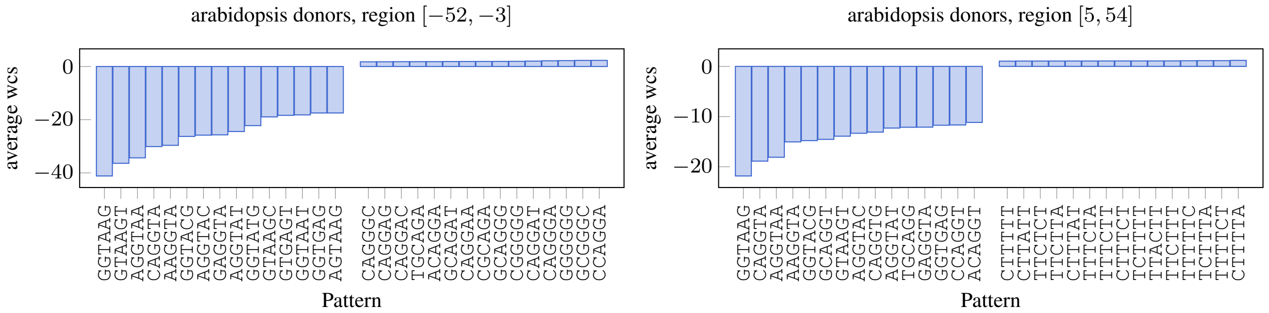
Fig. 3: The average weighted contribution score for each nucleotide, per position. For comparison purposes, we also make use of sequence logos to visualize the well-known splice site motifs, as determined by Stephens and Schneider (1992).



arabidopsis donors

arabidopsis acceptors

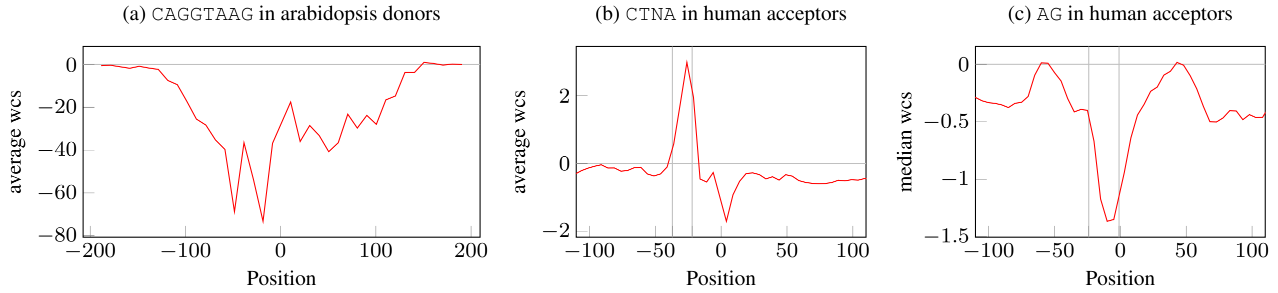
Fig. 4: Median weighted contribution scores for the different types of nucleotides at each position, for both arabidopsis donors and acceptors.



arabidopsis donors, region $[-52, -3]$

arabidopsis donors, region $[5, 54]$

Fig. 5: The lowest and highest average weighted contribution scores for patterns of length six occurring in the regions specified, for our arabidopsis donors model.



(a) `CAGGTAAG` in arabidopsis donors

(b) `CTNA` in human acceptors

(c) `AG` in human acceptors

Fig. 6: The average weighted contribution scores for occurrences of pattern `CAG`**`GT`**`AAGT` in arabidopsis donors, and of `CTNA` in human acceptors, and the median weighted contribution scores for occurrences of `AG` in human acceptors. On the x-axis, the position of the first nucleotide is indicated.

Funding

The research activities as described in this paper were funded by Ghent University Global Campus, Ghent University, imec, Flanders Innovation & Entrepreneurship (VLAIO), the Fund for Scientific Research-Flanders (FWO-Flanders), and the EU. Yvan Saeys is a Marylou Ingram Scholar.

References

Alipanahi, B., Delong, A., Weirauch, M. T., and Frey, B. J. (2015). Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nature Biotechnology*, **33**(8), 831–838.

Amit, M., Donyo, M., Hollande, D., Goren, A., Kim, E., Gelfman, S., Lev-Maor, G., Burstein, D., Schwartz, S., Postolsky, B., Pupko, T., and Ast, G. (2012). Differential GC content between exons and introns establishes distinct strategies of splice-site recognition. *Cell Reports*, **1**, 543–556.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, **10**(7), 1–46.

Bari, A. G., Reaz, M. R., and Jeong, B.-S. (2014). Effective DNA encoding for splice site prediction using SVM. *MATCH Communications in Mathematical and in Computer Chemistry*, **27**, 241–258.

Burset, M., Seledtsov, I. A., and Solovyev, V. V. (2001). Splicedb: database of canonical and non-canonical mammalian splice sites. *Nucleic Acids Research*, **29**(1), 255–259.

Dean, V., Delong, A., and Frey, B. (2016). Deep learning for branch point selection in RNA splicing. In *NIPS Workshop on Machine Learning in Computational Biology (MLCB)*.

Degroeve, S., Saeys, Y., De Baets, B., Rouzé, P., and Van de Peer, Y. (2005). SpliceMachine: predicting splice sites from high-dimensional local context representations. *Bioinformatics*, **21**(8), 1332–1338.

Foissac, S., Gouzy, J., Rombauts, S., Mathé, C., Amselem, J., Sterck, L., Van de Peer, Y., and Rouzé, P. (2008). Genome annotation in plants and fungi: Eugene as a model platform. *Current Bioinformatics*, **3**, 87–97.

Gao, K., Masuda, A., Matsuura, T., and Ohno, K. (2008). Human branch point consensus sequence is yUnAy. *Nucleic Acids Research*, **36**(7), 2257–2267.

Gooding, C., Clark, F., Wollerton, M. C., Grellscheid, S.-N., Groom, H., and Smith, C. W. (2006). A class of human exons with predicted distant branch points revealed by analysis of AG dinucleotide exclusion zones. *Genome Biology*, **7**(1), R1.

Iwata, H. and Gotoh, O. (2011). Comparative analysis of information contents relevant to recognition of introns in many species. *BMC Genomics*, **12**, 45.

Jian, X., Boerwinkle, E., and Liu, X. (2013). In silico tools for splicing defect prediction: a survey from the viewpoint of end users. *Genetics in Medicine*, **16**(7), 497–503.

Kim, Y. (2014). Convolutional neural networks for sentence classification. *Proceedings of EMNLP*, pages 1476–1751.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, **521**, 436–444.

Lee, T. and Yoon, S. (2015). Boosted categorical restricted boltzmann machine for computational prediction of splice junctions. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2483–2492.

Leung, M. K. K., Xiong, H. Y., Lee, L. J., and Frey, B. J. (2014). Deep learning of the tissue-regulated splicing code. *Bioinformatics*, **30**(12), i121–i129.

Nguyen, N. G., Tran, V. A., Ngo, D. L., Phan, D., Lumbanraja, F. R., Faisal, M. R., Abapihi, B., Kubo, M., and Satou, K. (2016). DNA sequence classification by convolutional neural network. *Journal of Biomedical Science and Engineering*, **9**, 280–286.

Paggi, J. and Bejerano, G. (2017). A sequence-based, deep learning model accurately predicts RNA splicing branchpoints. *bioRxiv preprint bioRxiv:185868*.

Pertea, M., Lin, X., and Salzberg, S. L. (2001). GeneSplicer: a new computational method for splice site prediction. *Nucleic Acids Research*, **29**(5), 1185–1190.

Quang, D. and Xie, X. (2016). DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of dna sequences. *Nucleic acids research*, **44**(11), e107.

Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3145–3153.

Sønderby, S. K., Sønderby, C. K., Nielsen, H., and Winther, O. (2015). Convolutional LSTM networks for subcellular localization of proteins. In *Proceedings of the Second International Conference on Algorithms for Computational Biology*, volume 9199, pages 68–80.

Sonnenburg, S., Schweikert, G., Philips, P., Behr, J., and Rätsch, G. (2007). Accurate splice site prediction using support vector machines. *BMC Bioinformatics*, **8**(Supplement 10), 1–16.

Stephens, R. M. and Schneider, T. D. (1992). Features of spliceosome evolution and function inferred from an analysis of the information at human splice sites. *Journal of Molecular Biology*, **228**(4), 1124–1136.

Sterck, L., Billiau, K., Abeel, T., Rouze, P., and Van de Peer, Y. (2012). Orcae: online resource for community annotation of eukaryotes. *Nature Methods*, **9**, 1041.

Trapnell, C., Pachter, L., and Salzberg, S. L. (2009). Tophat: discovering splice junctions with RNA-seq. *Bioinformatics*, **25**(9), 1105–1111.

Wang, K., Singh, D., Zeng, Z., Coleman, S. J., Huang, Y., Savich, G. L., He, X., Mieczkowski, P., Grimm, S. A., Perou, C. M., MacLeod, J. N., Chiang, D. Y., Prins, J. F., and Liu, J. (2010). MapSplice: accurate mapping of RNA-seq reads for splice junction discovery. *Nucleic acids research*, **38**(18), e178.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. pages 818–833.

Zeng, H., Edwards, M. D., Liu, G., and Gifford, D. K. (2016). Convolutional neural network architectures for predicting DNA-protein binding. *Bioinformatics*, **32**(12), i121–i127.

Zhang, Y., Liu, X., MacLeod, J. N., and Liu, J. (2016). DeepSplice: Deep classification of novel splice junctions revealed by rna-seq. In *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 330–333.