Wang *et al. BMC Bioinformatics* 2019, **20**(Suppl 23):652
https://doi.org/10.1186/s12859-019-3306-3

BMC Bioinformatics

METHODOLOGY Open Access

SpliceFinder: ab initio prediction of splice sites using convolutional neural network

Ruohan Wang, Zishuai Wang, Jianping Wang* and Shuaicheng Li*

From Joint 30th International Conference on Genome Informatics (GIW) & Australian Bioinformatics and Computational Biology Society (ABACBS) Annual Conference
Sydney, Australia. 9-11 December 2019

Abstract

Background: Identifying splice sites is a necessary step to analyze the location and structure of genes. Two dinucleotides, GT and AG, are highly frequent on splice sites, and many other patterns are also on splice sites with important biological functions. Meanwhile, the dinucleotides occur frequently at the sequences without splice sites, which makes the prediction prone to generate false positives. Most existing tools select all the sequences with the two dimers and then focus on distinguishing the true splice sites from those pseudo ones. Such an approach will lead to a decrease in false positives; however, it will result in non-canonical splice sites missing.

Result: We have designed SpliceFinder based on convolutional neural network (CNN) to predict splice sites. To achieve the *ab initio* prediction, we used human genomic data to train our neural network. An iterative approach is adopted to reconstruct the dataset, which tackles the data unbalance problem and forces the model to learn more features of splice sites. The proposed CNN obtains the classification accuracy of 90.25%, which is 10% higher than the existing algorithms. The method outperforms other existing methods in terms of area under receiver operating characteristics (AUC), recall, precision, and F1 score. Furthermore, SpliceFinder can find the exact position of splice sites on long genomic sequences with a sliding window. Compared with other state-of-the-art splice site prediction tools, SpliceFinder generates results in about half lower false positive while keeping recall higher than 0.8. Also, SpliceFinder captures the non-canonical splice sites. In addition, SpliceFinder performs well on the genomic sequences of *Drosophila melanogaster*, *Mus musculus*, *Rattus*, and *Danio rerio* without retraining.

Conclusion: Based on CNN, we have proposed a new *ab initio* splice site prediction tool, SpliceFinder, which generates less false positives and can detect non-canonical splice sites. Additionally, SpliceFinder is transferable to other species without retraining. The source code and additional materials are available at https://gitlab.deepomics.org/wangruohan/SpliceFinder.

Keywords: Canonical and non-canonical splice sites, Splice site prediction, Convolutional neural network

*Correspondence: jianwang@cityu.edu.hk; shuaicli@cityu.edu.hk
Department of Computer Science, City University of Hong Kong, 83 Tat Chee Ave, Kowloon Tong, Hong Kong, China

Background

Introduction

In recent years, high-throughput sequencing technologies have generated a large volume of genome sequences, which poses both opportunities and challenges to the identification of gene structure in genomes. The analysis of gene structure becomes one of the essential tasks in bioinformatics. A complete gene structure annotation includes the start codons, splice sites which are the boundaries between exons and introns, and stop codons. Many *in silico* methods are proposed to identify the aforementioned functional sites [1]. The success of an annotation system relies on accurate prediction of each component. In this work, we focus on the prediction of splice sites where accurate localization of splice sites can substantially help explore the structure of genes [2]. Furthermore, accurate prediction of splice sites can setup the boundaries of exons which is critical in alternative splicing prediction.

There are two types of splice sites, *donor* sites and *acceptor* sites where *donor* sites are located at the junction of exon-intron and *acceptor* sites mark the intron-exon boundaries. Two highly conserved dinucleotides are observed on the splice sites, GT for *donor* sites and AG for *acceptor* sites [3, 4]. The splice sites confirming the GT-AG consensus are called canonical splice sites.

We now introduce the main factors which affect the accuracy of splice site prediction. Firstly, the existence of the dinucleotide GT or AG is not necessary for identifying the splice sites, some non-canonical splice sites without the dimers may be observed [5–9]. Though non-canonical splice sites may not appear frequently [10, 11], some of them are vital in immunoglobulin gene expression and other important biological events [11]. Secondly, the existence of the dinucleotide GT or AG is not a sufficient condition for splice sites since dimers frequently occur at the sequences that are not splice sites. In this paper, we address the aforementioned two issues in splice site prediction.

Related work

The existing splice site prediction tools work on data from either RNA sequences or DNA sequences. For RNA-seq based tools, TopHat [12], SpliceMap [13], and MapSplice [14] apply the alignment-based approach by mapping the reads from RNA-seq experiments to the reference genome and discovering the exon-exon junctions. The alignment-based approach makes it easier to avoid false positives since it relies on the junction signals, instead of patterns. However, the need for a reference genome limits its application. Recently, deep neural networks have been employed to predict splice sites from arbitrary pre-mRNA transcript sequences [15].

For tools based on DNA sequences, they utilized learning models to learn the features around splice sites. With more advanced machine learning algorithms designed, complex patterns are likely to be learned, which leads to the improvement in prediction accuracy. For example, GeneSplicer applies the decision tree algorithm and enhances it with Markov models to capture additional information around splice sites [16]. SpliceMachine employs linear support vector machines to build a linear model, in order to predict splice sites from high-dimensional local context representations [17]. Support vector machines with weighted degree kernel have also been applied to genome-wide predictions of splice sites [18]. In recent years, deep networks have been widely utilized to detect splice signals from genomic data. A novel deep belief network with restricted Boltzmann machines training method has been proposed for the class-imbalanced problem [19] in splice site prediction. Long short-term memory (LSTM) [20] and convolutional neural networks (CNN) [21] have also been tried to improve the performance. However, the learning models have the shortcoming of excessive false positives. To solve the problem, most tools firstly choose all the sequences with canonical signals (GT for *donor* sites and AG for *acceptor* sites) as candidate splice sites and then distinguish between true splice sites and pseudo splice sites [16–18, 21]. In spite of the decrease of false positives, these tools would miss all the non-canonical splice sites.

Based on the existing problems of splice site prediction, we propose to design a splice site prediction tool, named SpliceFinder, which has the following strengths:

(i) The model is trained with genomic data directly, so it can achieve the *ab initio* prediction of splice sites.
(ii) Not only canonical but also non-canonical splice sites can be predicted correctly.
(iii) The number of false positives decreased since SpliceFinder considers more information besides AG or GT pattern to identify splice sites.

Methods

Datasets

DNA sequence data (FASTA files) and annotations of the corresponding sequences (GTF files) were downloaded from Ensembl [22]. Our models were trained using human reference genome (*GRCh38*). Since the reverse strand is the reverse-complementary strand of the forward strand, we only considered the forward strand. The hg38 dataset contains 29742 genes with 21 exons per gene on average, most of these exons have duplicates due to alternative splicing [23]. We randomly chose a certain number of exons to generate training set for *donor* sites and *acceptor* sites.

For the purpose of testing the models on other species, we also downloaded the genomic sequences of *Drosophila melanogaster* (*BDGP6*), *Mus musculus* (*GRCm38*), *Rattus* (*Rnor_6.0*), and Danio rerio (*GRCz11*) from Ensembl.

Convolutional neural network

Neural networks (NNs) consist of connections between neurons, NNs learn from dataset by adjusting the weights of the connections. However, the weights for different positions are independent, NN is not enough for finding the particular patterns of splice sites over the sequences [21]. Therefore, a convolutional layer, which enables shared weights, is added to the NN [24]. Figure 1 and the following descriptions provide a summary of the input and architecture of our CNN.

Before training, the sequence data are transformed using one-hot encoding. A (Adenine) is encoded as (1 0 0 0), C (Cytosine) is encoded as (0 1 0 0), G (Guanine) is encoded as (0 0 1 0), T (Thymine) is encoded as (0 0 0 1), and N (uncertain nucleic acid) is encoded as (0 0 0 0). As a result, each sequence can be represented with a $L \times 4$ matrix where L is the length of the sequence and 4 is the size of nucleotides vocabulary. The encoded sequence is the input of our neural network.

The first layer of our neural network is a convolutional layer. The genomic sequence is considered as a 1-D sequence window with a fixed length L and four channels (A, C, G, T). The convolutional layer is supposed to extract the pattern information with 50 kernels of size 9, and the length of scanning step is set to 1, for the purpose of preserving the integrity of genetic code. The output of the convolutional layer is a $L \times 50$ feature map, where L is the length of the sequence. Different numbers of convolutional layers were tried and the results are shown in Additional file 1. The NN with one convolutional layer gives the best performance. Consequently, the following experiments will apply one convolutional layer.

The following layer is a fully connected layer with 100 neurons. The fully connected layer is employed in order to improve the nonlinear expression ability of our neural network, so that the model is more likely to detect those non-canonical splicing signals. ReLU [25] is applied as the activation function in this layer. In order to avoid overfitting, a dropout layer [26] is used to randomly mask out 30% of the output. The final fully connected layer has three neurons which correspond to *acceptor* site, *donor* site, and non-splice-site. Softmax activation function [27] is used for the neurons in the last fully connected layer to convert the output into normalized probability.

For training, cross-entropy [28] is used as the loss function, and Adam algorithm [29], with the learning rate of 10^{-4}, is applied for optimization. The number of epochs for training is set to 40, with the batch size of 50. We used Keras Python package to build our CNN.

Training and testing procedure

The human reference genome (*GRCh38*) was used to construct the dataset. We obtained the location of exons from the annotation file, took sequences centered at the right and left boundaries of exons, which correspond to *donor* sites and *acceptor* sites, as the positive set of our training data, and then we took sequences centered at the intermediate position of two adjacent splicing sites as the initial negative set. The dataset contains 10000 sequences

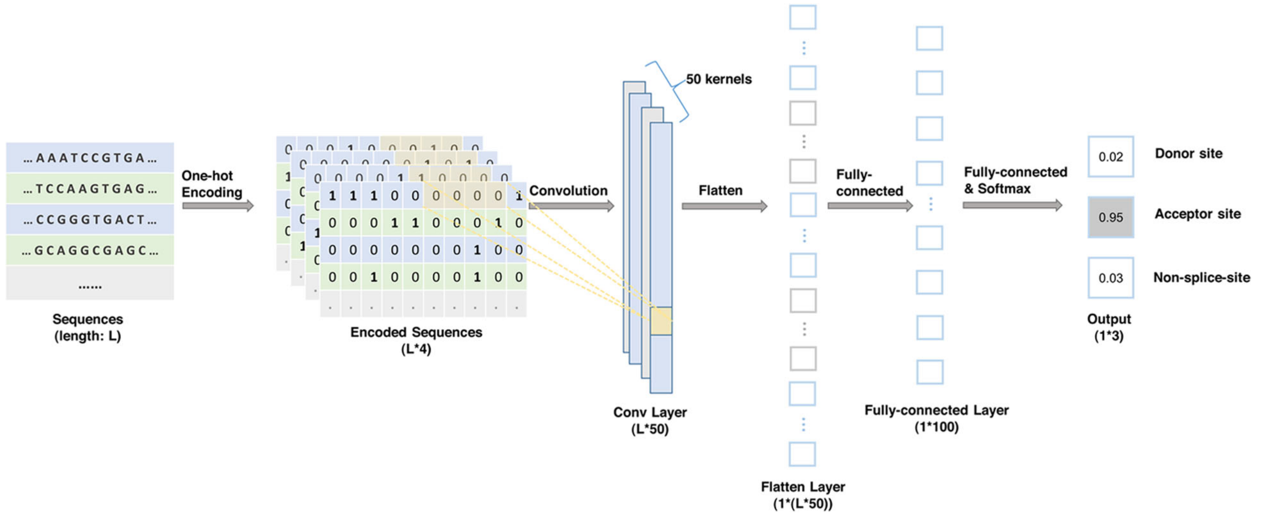


Fig. 1 The architecture of our proposed CNN. The input of the neural network is the encoded DNA sequence with the length of L. The first layer is a 1-D convolutional layer, consists of 50 kernels with the size of 9. The second layer is a fully connected layer with 100 neurons, followed by a dropout layer. Another fully connected layer and softmax activation function are applied for the final prediction

of *donor* site, *acceptor* site, and non-splice-site, which are randomly selected. Among the 30000 sequences, we used 90% for training and 10% for testing, and then 20% of the training data were set as validation data, which were used for checking network structure, hyper-parameters, and sequence length.

Next, we used our trained model to predict the splice sites on real-world long genomic sequences. Now that the model requires fixed length input, we decided to use a sliding window to detect every location on the long sequences. Only when the splicing site locates in the middle of an input sequence will our model give a positive classification result. Therefore, by moving the sliding window along the sequence and putting every subsequence into our model, we will get the exact positions of every splicing site. To decrease the number of false positives, we also used two classifiers for those subsequences considered to have splice sites. The structures of these two classifiers are the same as the structure of our proposed CNN, except that the last fully connected layer has only 2 neurons, to predict whether it is a *donor* (*acceptor*) site or false positive.

Dataset reconstruction

In spite of the high classification accuracy, we found that when we apply the model to a long real-world sequence, most non-canonical splice sites could not be found out, and there were still many false positives. We believe the major cause is that the training set is not comprehensive. To solve the unbalance problem of the dataset, firstly, we reconstructed the positive set by achieving a canonical: non-canonical ratio of 10:1 (totally 22000 sequences). Then, for the negative set, there are too many sequences containing GT or AG that are not annotated as splice sites, so we used an iterative approach [30] to reconstruct the negative set:

(i) We used 90% of the current dataset as the training set. The model was trained with the training set. The rest of the dataset was used to test the performance of our CNN on the current dataset.

(ii) The model was used to predict splice sites on the sequence of a randomly chosen human gene with a sliding window.

(iii) The false positives generated from the last step were added to the negative set. The new negative set, together with the positive set, constituted the new dataset.

This procedure was repeated until the number of false positives did not decrease anymore when testing a gene sequence which is set aside in advance. These steps are illustrated in Fig. 2. The method forces the convolution neural network to learn more features for the classification task.

Performance evaluation

Our models can be applied to discriminating short sequences and predicting splice sites on long sequences. For short sequences classification, we measured performance in terms of accuracy, area under receiver operating characteristics curve (AUC), recall, precision, and F1 score:

$$Recall = \frac{TP}{TP + FN},$$

$$Precision = \frac{TP}{TP + FP},$$

$$F1\ score = (\frac{Recall^{-1} + Precision^{-1}}{2})^{-1}.$$

Since all the measures except accuracy are applicable for two-class classification, and other methods used for comparison are designed only for two-class problems, we calculated AUC, recall, precision, and F1 score for *donor* site and *acceptor* site separately. The ROC curves and precision-recall curves were also made for the performance evaluation.

For long genomic sequences, on the one hand, most of the existing tools can not find all the splice sites no matter how to set the parameters; on the other hand, the prediction with low recall is meaningless even though the number of false positives is small. In order to consider both the value of recall and the number of false positives, we counted the number of false positives when 100% and 80% of the splice sites are successfully predicted separately.

Results

Testing different length of input

To choose the most suitable region for training, we used sequences of different lengths as the input of our models. Since the sequence lengths used in other splice site prediction tools range from 40 to 400 nt, we varied the input lengths in this range. Using the initial dataset, we found that all the test accuracies reach 95%. However, the accuracies have significant declines after the iterative process because of the increased complexity of test data. For instance, the accuracy decreases from 96.9% to 83.2% for sequence length of 40 nt, and for sequence length of 400 nt, the accuracy changes from 96.5% to 90.3%. Longer sequences help models keep good performance since they provide more information. As shown in Fig. 3, CNN achieves the best performance with the input length of 400 nt for the reconstructed dataset. Accordingly, subsequent experiments will use this length of input.

Decomposing the output of CNN

To improve the interpretability of our neural network, we utilized DeepLIFT to analyze the contributions of different regions inside the sequence window to the output.

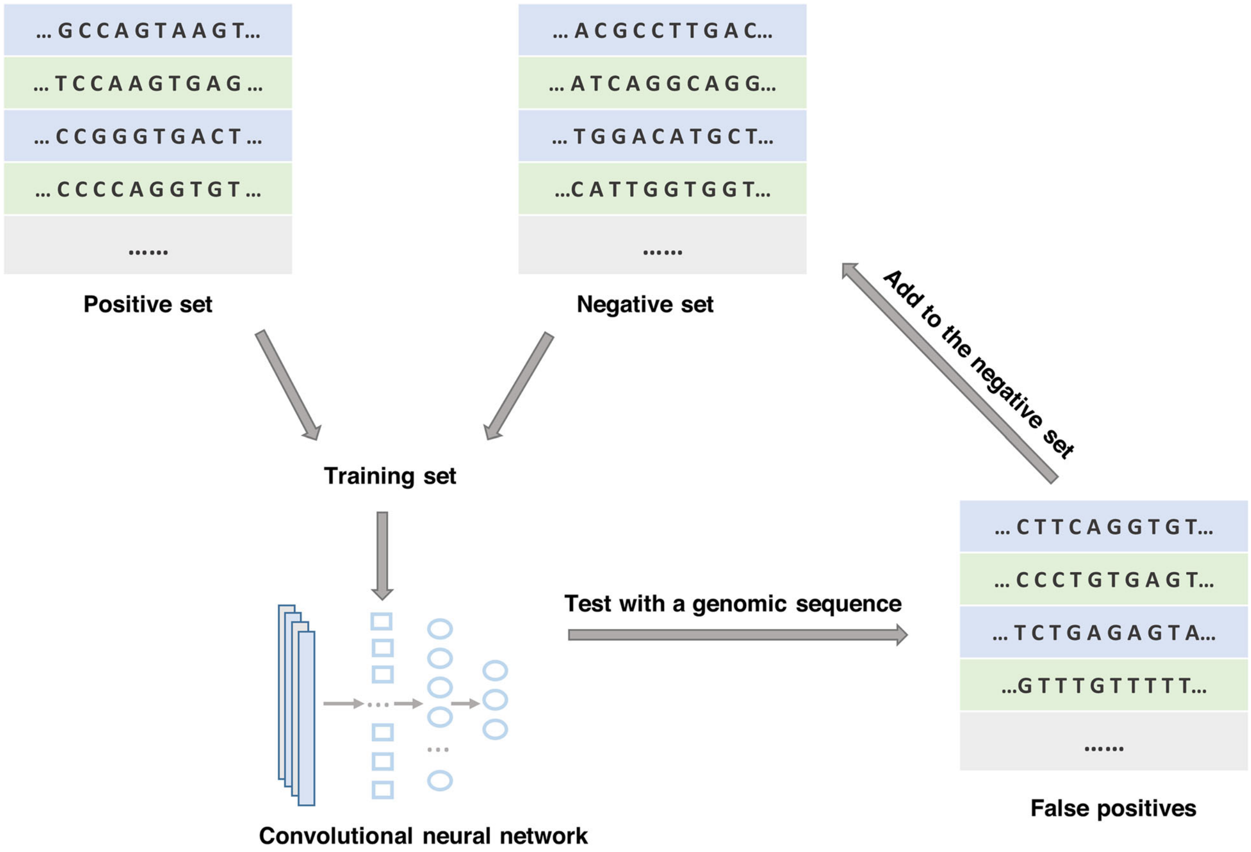

... G C C A G T A A G T...
... T C C A A G T G A G ...
... C C G G G T G A C T ...
... C C C C A G G T G T ...
......
Positive set
... A C G C C T T G A C...
... A T C A G G C A G G...
... T G G A C A T G C T...
...C A T T G G T G G T...
......
Negative set
Add to the negative set
Training set
Test with a genomic sequence
... C T T C A G G T G T...
... C C C T G T G A G T...
... T C T G A G A G T A...
...G T T T G T T T T T...
......
False positives
Convolutional neural network

Fig. 2 The iterative approach for negative set reconstruction. At each iteration, the trained CNN is tested with a randomly chosen genomic sequence, the false positives are collected and added to the training data, which will be used to train our CNN at the next iteration

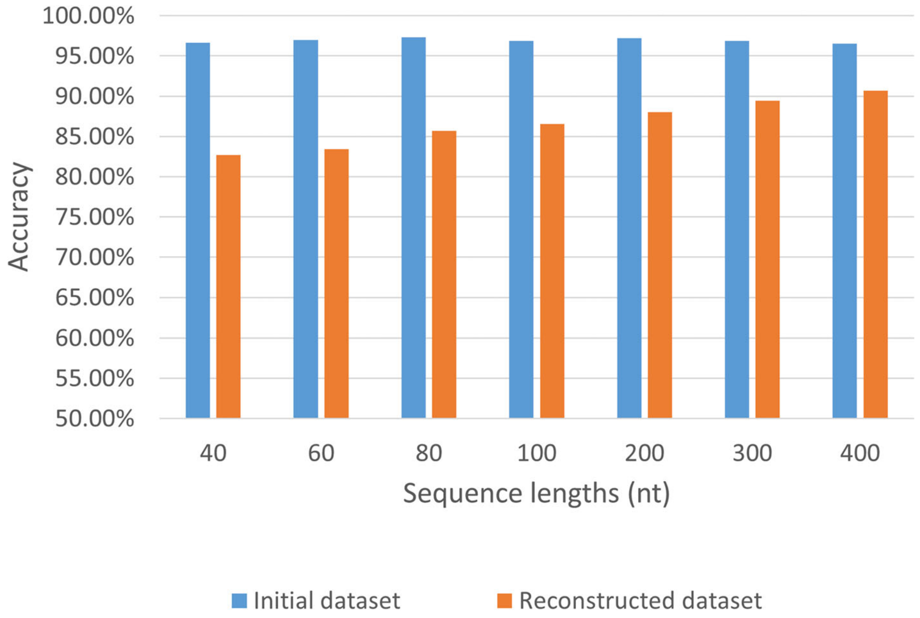


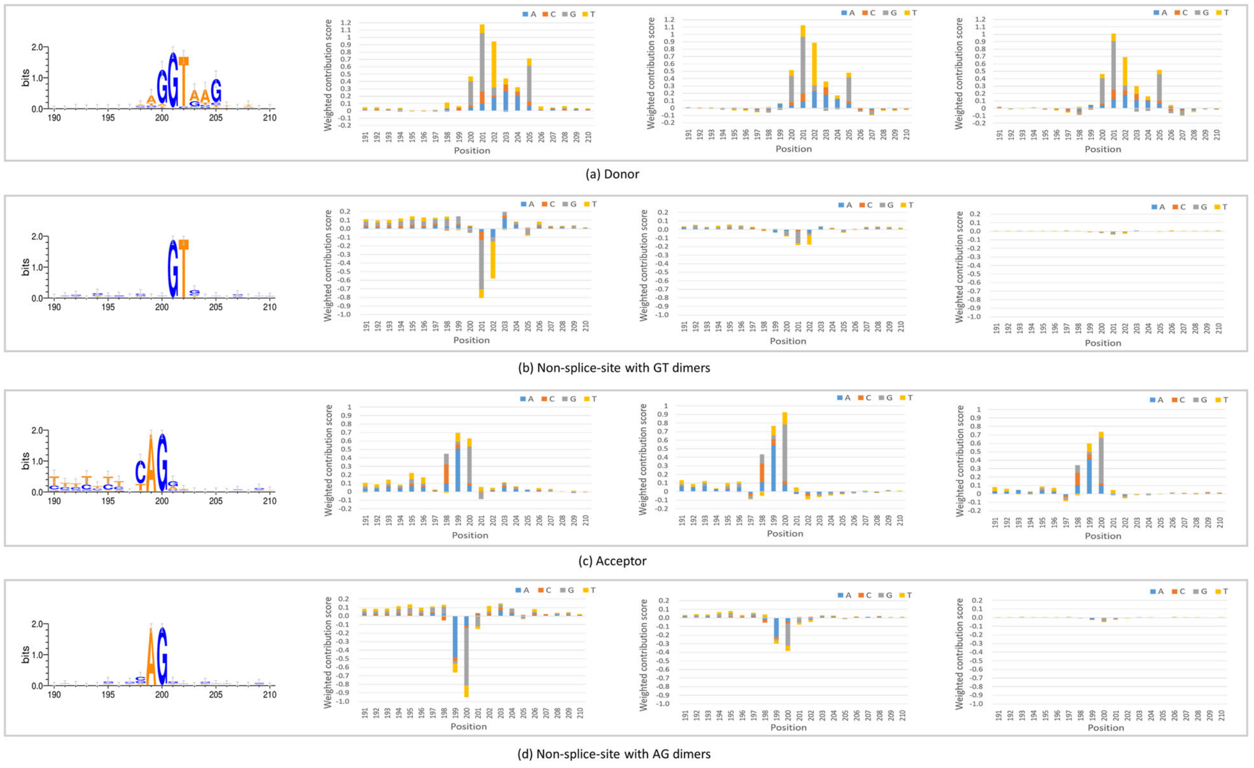
Fig. 3 The effect of sequence length on accuracy. Varying the sequence lengths from 40 to 400 nt, the classification accuracies for the test set of initial dataset and reconstructed dataset are compared

DeepLIFT uses the backpropagation method to decompose the prediction of neural network on the input and computes the weighted contribution scores for every part of the input, thus making neural network no longer a “black box” [31]. We randomly chose 100 sequences with *donor* sites, 100 sequences with *acceptor* sites, 100 non-splice-site sequences with GT dimers, and 100 non-splice-site sequences with AG dimers, from human reference genome, and then applied three models generated from different periods of the iterative process to classifying these sequences. The average weighted contribution scores for 20 nucleotides near the splice sites are computed, which show the influence of each nucleotide on the right decision. A positive score means a positive role in making the right decision while a negative score means an opposite effect. The result is shown in Fig. 4. For sequences with *donor* site or *acceptor* site, the contribution scores of GT or AG are high for all of the three models. However, for non-splice-site sequences with the dinucleotide GT or AG, the negative influence of the two dimers decreases a lot for the models generated from later periods of the iterative process.

Comparison of classification performance

We compared our CNN with common machine learning algorithms (Logistic regression [33], Decision tree [34], Random forest [35], SVM [36] with linear and RBF kernel), DBN [37], and LSTM [38], the last two algorithms have been applied to predicting splice junctions previously and all the parameters are set as described by the authors [19, 20]. We trained and tested the mentioned algorithms using the reconstructed dataset. There are also tools that require more information besides genomic sequences, like RNA sequences, or only accept long sequences as inputs. For the second case, we will compare these tools with our models on long genomic sequences later.

Compared with other algorithms, SpliceFinder has the best performance with regard to all the measures. Figure 5 (a) presents the comparison of accuracies between SpliceFinder and other machine learning algorithms. The classification accuracy of SpliceFinder exceeds 90% while the accuracies of all other machine learning algorithms do not reach 80%. Figure 5 (b) and (c) show the ROC curves and precision-recall curves for *donor* site and *acceptor* site respectively. SpliceFinder achieves the largest areas under



(a) Donor

(b) Non-splice-site with GT dimers

(c) Acceptor

(d) Non-splice-site with AG dimers

Fig. 4 The sequence logos and average weighted contribution scores of nucleotides near the splice site. For *donor* sites, *acceptor* sites, and non-splice-sites with canonical signals, the average weighted contribution scores of different models for each nucleotide near the splice site (located at the position between 200 and 201) is shown. From left to right, the models are generated from the 1st, 50th, and 100th iteration. The sequence logos are made [32] to show the difference of patterns between true and false splice sites. **a** Donor. **b** Non-splice-site with GT dimers. **c** Acceptor. **d** Non-splice-site with AG dimers

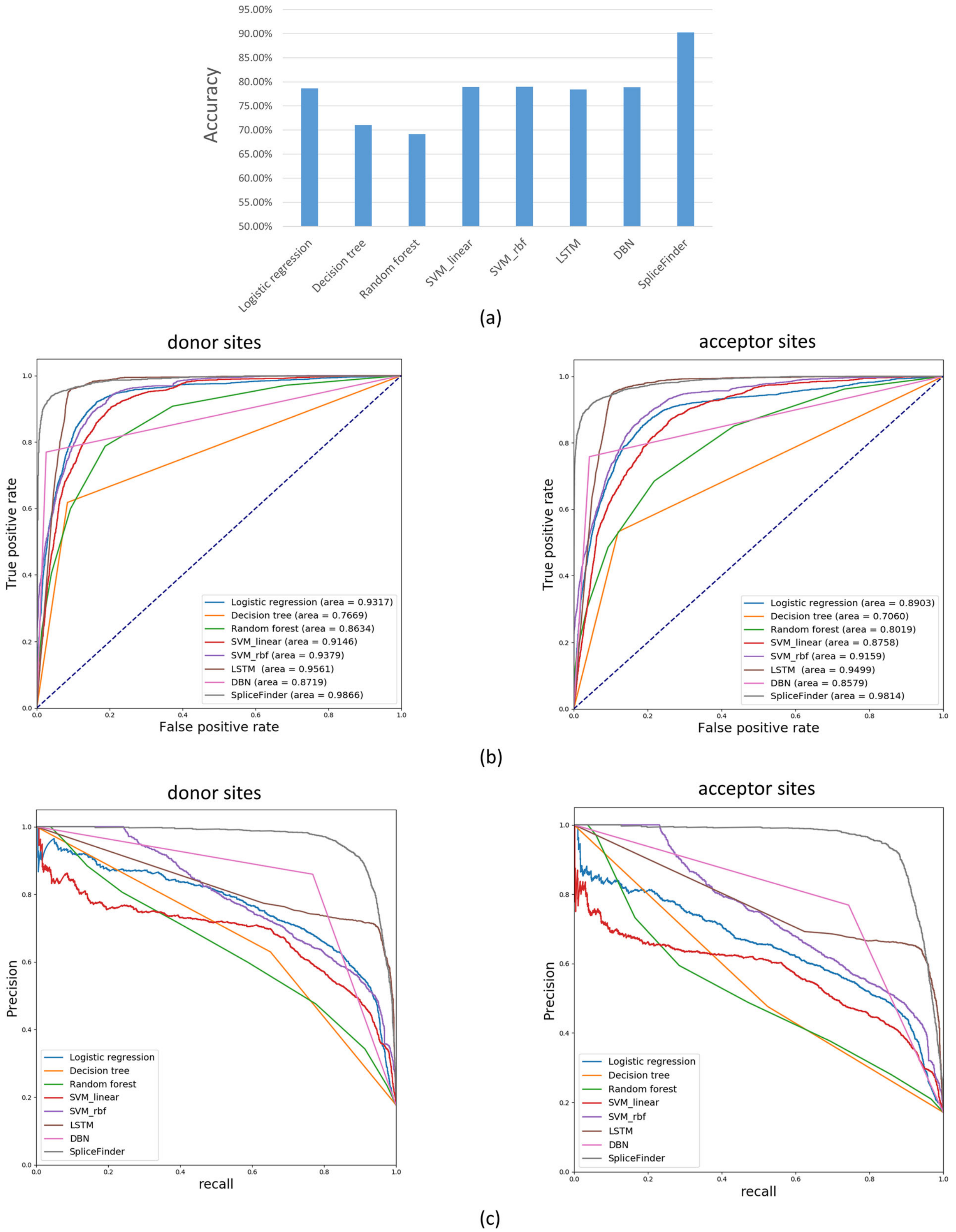


Fig. 5 Comparison of classification performance of different methods on the test set of the reconstructed dataset. The compared measures include (**a**) classification accuracy; (**b**) ROC curve for *donor* sites (left) and *acceptor* sites (right); (**c**) Precision-recall curve for *donor* sites (left) and *acceptor* sites (right)

the curves apparently. Additionally, in terms of recall, precision, and F1 score, SpliceFinder obtains higher scores than other algorithms, the comparison result can be found in Additional file 2.

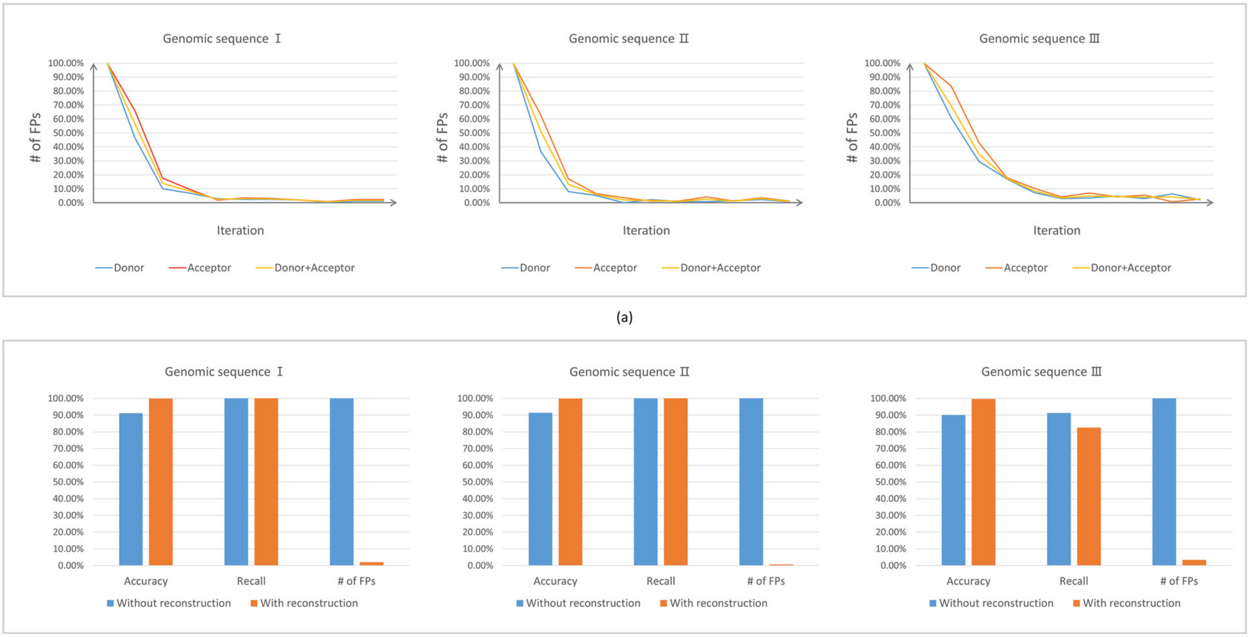
Prediction performance on genomic sequences

To evaluate our method and compare it with other tools, we used SpliceFinder to predict the splice sites on three randomly chosen human genomic sequences. These sequences are set aside in the iterative process.

We employed the models generated in the iterative process and found the numbers of false positives have sharp declines with iteration, shown in Fig. 6 (a). Since the numbers of false positives decrease and the values of recall remain high, the model obtains better accuracies on the three genomic sequences after dataset reconstruction (Fig. 6 (b)). Next, we compared the performance of SpliceFinder with GeneSplicer [16], SpliceMachine [17], and SpliceRover [21]. When recall is 1, the number of false positives was calculated. Since some tools can not reach the recall of 1 at all, we also compared the number of false positives when recall is over 0.8. As shown in Table 1, with recall reaching 1 and 0.8, SpliceFinder generated the least false positives. During the experiment, we noticed that when the score cutoff was set to be 0, SpliceRover would consider all the sequences with GT patterns as *donor* sites and all the sequences with AG patterns as *acceptor* sites. However, Fig. 7 shows that SpliceRover still misses a *donor* site even with the score cutoff set as 0 for Genomic Sequence III. Therefore, there is a *donor* site without GT pattern on this sequence. Despite this, for SpliceFinder, the recall of models generated from early iteration is 1, which means the models have the ability to find the non-canonical splice sites.

Testing on other species

Since GT-AG rule can be applied to the splice sites of all eukaryotic genes, we also used our trained models to predict splice sites on the genomic sequences of *Drosophila melanogaster*, *Mus musculus*, *Rattus*, and *Danio rerio*. 3 long sequences for each species were chosen randomly. Shown in Fig. 8, with iteration, the models obtained higher accuracies. Since high accuracy is not enough for our task, we also calculated the numbers of false positives and values of recall (See Fig. 9). For convenience, both *donor* site and *acceptor* site are considered as the positive set, so the recall in Fig. 9 is the percentage of splice sites to be predicted successfully. It can be seen that with iteration, the models are tended to give less false positives, but keep the recall higher than 0.8 for the 12 genomic sequences of different species in Fig. 9. Other mentioned machine learning algorithms are also be compared on other species, SpliceFinder achieves the best performance on the four species (Additional file 3). Additionally, we found the model trained with *Homo sapiens* data have the same performance as the model trained with other species



(a)

(b)

Fig. 6 The prediction performance improves after dataset reconstruction. **a** Using the models generated in the iterative process to predict the splice sites on three randomly chosen genomic sequences, false positive numbers of both *donor* site and *acceptor* site are shown. The false positive numbers of the initial model are set as 100%. **b** The comparison of accuracy, recall, and false positives numbers between models with and without dataset reconstruction

Table 1 Comparison of prediction performances of different softwares on three randomly chosen genomic sequences

(a)

Genomic Sequence I	*Donor*	*Acceptor*	*Donor & Acceptor*
GeneSplicer	N/A	N/A	N/A
SpliceMachine	159	62	221
SpliceRover	19	16	35
SpliceFinder	7	5	**12**
Genomic Sequence II	*Donor*	*Acceptor*	*Donor & Acceptor*
GeneSplicer	N/A	N/A	N/A
SpliceMachine	72	54	126
SpliceRover	3	6	9
SpliceFinder	1	2	**3**
Genomic Sequence III	*Donor*	*Acceptor*	*Donor & Acceptor*
GeneSplicer	N/A	N/A	N/A
SpliceMachine	N/A	N/A	N/A
SpliceRover	N/A	N/A	N/A
SpliceFinder	24	35	**59**

(b)

Genomic Sequence I	*Donor*	*Acceptor*	*Donor & Acceptor*
GeneSplicer	N/A	N/A	N/A
SpliceMachine	159	62	221
SpliceRover	19	3	22
SpliceFinder	5	5	**10**
Genomic Sequence II	*Donor*	*Acceptor*	*Donor & Acceptor*
GeneSplicer	9	4	13
SpliceMachine	10	4	14
SpliceRover	0	3	**3**
SpliceFinder	1	2	**3**
Genomic Sequence III	*Donor*	*Acceptor*	*Donor & Acceptor*
GeneSplicer	N/A	N/A	N/A
SpliceMachine	20	90	110
SpliceRover	444	21	465
SpliceFinder	6	6	**12**

The numbers of false positives when recall reaches 1 (a) and 0.8 (b) are shown. (Note: N/A implies the software can not reach the recall of 1 or 0.8, no matter how to set the parameters. The best performance is in bold.)

data or multiple species data, on the genomes of all the four species (Additional file 4).

Discussion

Analyzing the data reconstruction

The initial dataset is randomly generated from human genomic sequences. In this case, only a small number of sequences without splice sites contain the GT or AG pattern, while almost all the splice sites confirm the GT-AG consensus. Using this simple feature, many algorithms have good performance in the classification tasks even with the input of short length. However, when being applied to the prediction task on long genomic sequences, we found that our models still miss those non-canonical splice sites. More seriously, any sequence with GT or AG pattern is easily misclassified as *donor* site or *acceptor* site, which leads to a large number of false positives. Most tools also have these two problems. To decrease the number of false positives, many existing splice site prediction algorithms transfer the problem to two classification

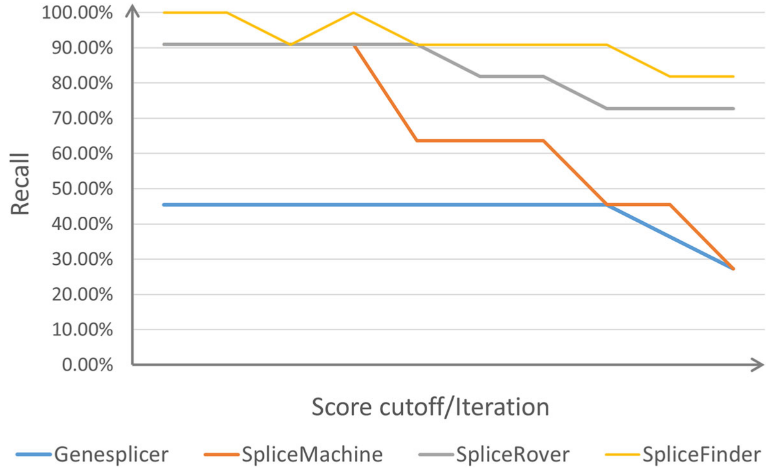
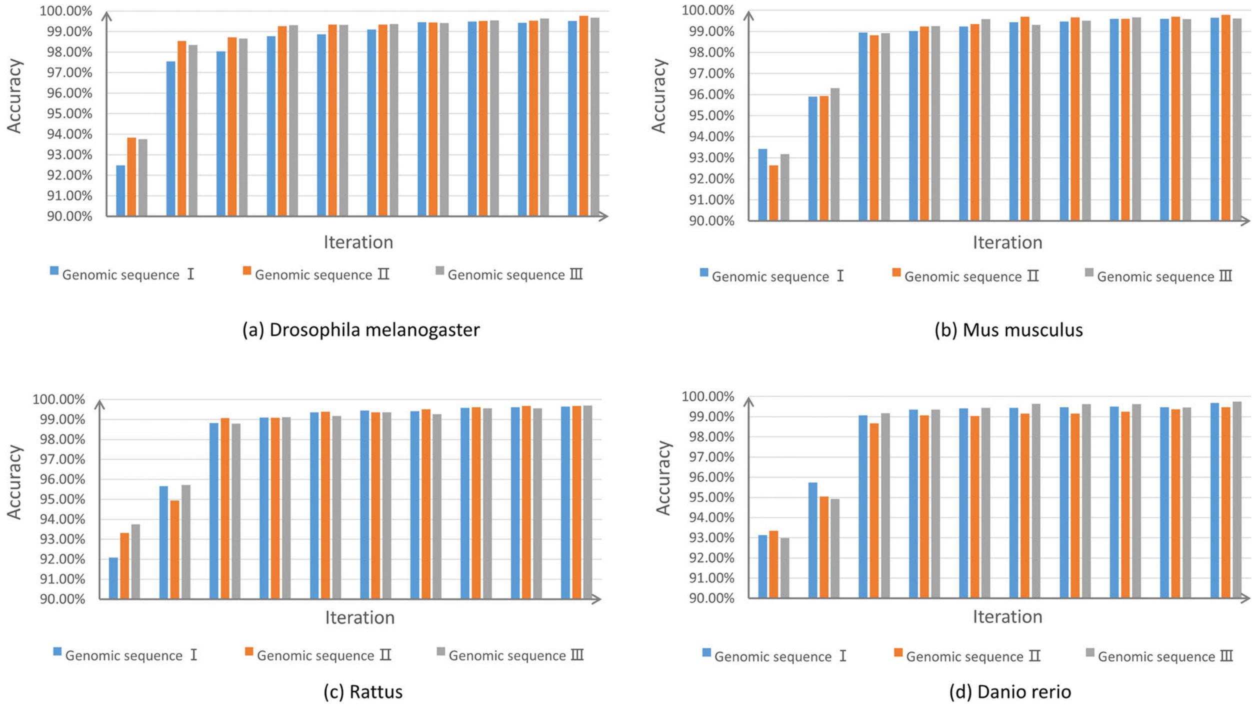


Fig. 7 Comparison of recall of different softwares for *donor* sites of Genomic Sequence III. Using different score cutoff or models generated in the iterative process, the recall values of the four softwares, for *donor* sites of Genomic Sequence III, are calculated

problems: discriminating sequences with splice sites from sequences without splice sites but with the dinucleotide GT or AG for *donor* and *acceptor* sites separately [16–18, 21]. However, their methods focus on canonical splice sites, only sequences with the consensus GT or AG will be classified, which will definitely miss non-canonical splice sites.

Therefore, we decided to consider the problem in a different way. We think the most important cause for the missed splice sites and false positives is the unbalance of the training dataset, With the simple training set, the classifier cannot learn enough information to find non-canonical splice sites and exclude non-splice-sites with canonical signals. Increasing the proportion of non-canonical splice sites and the iterative process make the training set cover more information, therefore the models can deal with those unusual cases better after the reconstruction of dataset. Although the classification accuracy



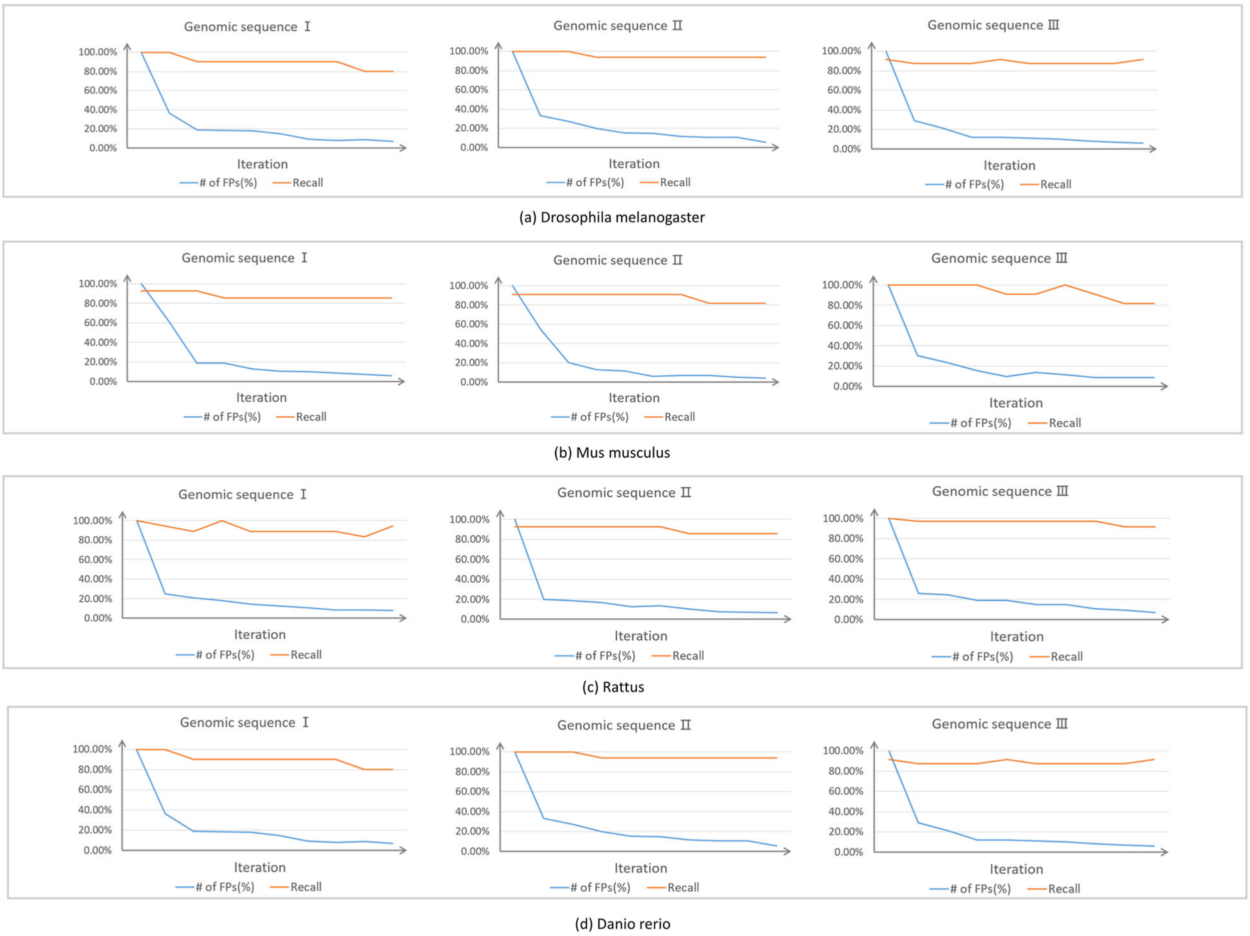
(a) Drosophila melanogaster

(b) Mus musculus

(c) Rattus

(d) Danio rerio

Fig. 8 The splice site prediction accuracy of our models for other species. For (**a**) *Drosophila melanogaster*, (**b**) *Mus musculus*, (**c**) *Rattus*, and (**d**) *Danio rerio*, the models generated in the iterative process are applied to predicting the splice sites on three randomly chosen genomic sequences



(a) Drosophila melanogaster

(b) Mus musculus

(c) Rattus

(d) Danio rerio

Fig. 9 The false positive numbers and recall of our models for other species. For (**a**) *Drosophila melanogaster*, (**b**) *Mus musculus*, (**c**) *Rattus*, and (**d**) *Danio rerio*, the numbers of false positive and values of recall are calculated to show more details of the prediction performance for other species

is decreased, the models have better prediction performance on long genomic sequences. Of course, since the information in the training set is much more difficult to learn, it requires longer sequences as input and models with stronger pattern-find ability.

Strengths of SpliceFinder

Based on the experimental results, we believe SpliceFinder has the following strengths:

(i) Trained with data generated directly from human genomic sequences, SpliceFinder has achieved *ab initio* splice site prediction. The experimental results have proved that SpliceFinder has good prediction performance using only genomic sequences.

(ii) To improve the sensitivity of our models for the non-canonical splice sites, we increased the number of splice sites without GT-AG patterns in the training data, so unlike other existing tools, SpliceFinder also considers non-canonical splice sites.

(iii) Instead of simply increasing the score cutoff, we used an iterative process to reconstruct the dataset, in order to decrease the number of false positives. We can see SpliceFinder makes less false positive predictions while still successfully finding the true splice sites.

(iv) SpliceFinder is a 3-class model, which can directly give the classification result of *donor* site, *acceptor* site or non-splice-site. Compared with other tools with 2-classes models, SpliceFinder is more straightforward and convenient to use.

(v) SpliceFinder can be used to predict splice sites on the genomic sequences of other species with no need to retrain, so it can be applied to the annotation of new species.

Future work

Our future work will continue to explore the following topics:

(i) Noticing that SpliceFinder can also be applied to other species without retraining, we plan to combine

SpliceFinder with other tools based on RNA-seq [12–14], SpliceFinder provides the locations of splice sites for reference, RNA-seq based tools use the alignment approach to identify the actual splice sites. We strongly believe the two methods will work together and play complementary roles, especially for new species without reference genome.

(ii) A complete annotation system needs to predict not only the splice sites but also the transcript start sites (TSSs) and transcription termination sites (TTSs) [39]. There are also conserved sequences around TSSs and TTSs, for example, the well-known TATA box is universally observed in the core promoter region [40]. However, how to handle the promoter sequences without TATA box and the non-promoter sequences with TATA patterns remains a problem. It is also a pattern-based task, so we plan to adjust the structure of our CNN and expand the application of our tool.

Conclusions

In this paper, we introduced a new tool for splice site prediction, named SpliceFinder. SpliceFinder applies convolutional neural network to classify sequences to *donor* site, *acceptor* site or non-splice-site. With a sliding window, it can predict the exact position of every splice site on long genomic sequences with less false positives and high recall. Compared with other splice site prediction tools, SpliceFinder has better prediction performance, it also has the ability to find the non-canonical splice sites. Additionally, the models trained with human genome are also applicable to other species without retraining, which makes SpliceFinder useful in the annotation of new species.

Supplementary information

Supplementary information accompanies this paper at https://doi.org/10.1186/s12859-019-3306-3.

Additional file 1: Figure S1 The performance of models with different number of layers.

Additional file 2: Figure S2 Evaluation of different methods with various metrics.

Additional file 3: Figure S3 The splice site prediction accuracy of different methods for other species.

Additional file 4: Figure S4 The performance of models trained with data of other species.

Abbreviations

AUC: Area under receiver operating characteristics curve; CNN: Convolutional neural network; LSTM: Long short-term memory; mRNA: Messenger RNA; NN: Neural network; Pre-mRNA: Precursor-messenger RNA; ROC: Receiver operating characteristics; TSS: Transcript start site; TTS: Transcript termination site

Acknowledgments

We would like to thank the Department of Computer Science, City University of Hong Kong for providing RTX 2080 Ti Graphics Card for us to do the experiments.

About this supplement

This article has been published as part of *BMC Bioinformatics Volume 20 Supplement 23, 2019: Proceedings of the Joint International GIW & ABACBS-2019 Conference: bioinformatics*. The full contents of the supplement are available online at https://bmcbioinformatics.biomedcentral.com/articles/supplements/volume-20-supplement-23.

Authors' contributions

RW designed the neural network, performed most experiments, and drafted the manuscript. ZW collected the data, helped performing the experiments, and revised the manuscript. JW supervised this project and revised the manuscript. SL proposed the topic, supervised this project, and revised the manuscript. All authors have read and approved of the final manuscript.

Funding

Publication costs are funded by Science Technology and Innovation Committee of Shenzhen Municipality under project JCYJ20170818095109386. The work described in this paper is also supported by a GRF Project grant from the RGC General Research Fund (9042181; CityU 11203115) and the GRF Research Project (9042348; CityU 11257316).

Availability of data and materials

The datasets analysed during the current study are available in the Ensembl repository, https://asia.ensembl.org/info/data/ftp/index.html.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Published: 27 December 2019

References

1. Rätsch G, Sonnenburg S, Srinivasan J, Witte H, Müller K-R, Sommer R-J, Schölkopf B. Improving the caenorhabditis elegans genome annotation using machine learning. PLoS Comput Biol. 2007;3(2):20.
2. Reese MG, Eeckman FH, Kulp D, Haussler D. Improved splice site detection in genie. J Comput Biol. 1997;4(3):311–23.
3. Breathnach R, Benoist C, O'hare K, Gannon F, Chambon P. Ovalbumin gene: evidence for a leader sequence in mrna and dna sequences at the exon-intron boundaries. Proc Natl Acad Sci. 1978;75(10):4853–7.
4. Mount SM. A catalogue of splice junction sequences. Nucleic Acids Res. 1982;10(2):459–72.
5. Hodge MR, Cumsky MG. Splicing of a yeast intron containing an unusual 5'junction sequence. Mol Cell Biol. 1989;9(6):2765–70.
6. Quan F, Forte MA. Two forms of drosophila melanogaster gs alpha are produced by alternate splicing involving an unusual splice site. Mol Cell Biol. 1990;10(3):910–7.
7. Jackson IJ. A reappraisal of non-consensus mrna splice sites. Nucleic Acids Res. 1991;19(14):3795.
8. Xue J, Rask L. The unusual 5' splicing border gc is used in myrosinase genes of the brassicaceae. Plant Mol Biol. 1995;29(1):167–71.
9. Pollard AJ, Krainer AR, Robson SC, Europe-Finner GN. Alternative splicing of the adenylyl cyclase stimulatory g-protein gαs is regulated by sf2/asf and heterogeneous nuclear ribonucleoprotein a1 (hnrnpa1) and involves the use of an unusual tg 3'-splice site. J Biol Chem. 2002;277(18):15241–51.
10. Burset M, Seledtsov I, Solovyev V. Analysis of canonical and non-canonical splice sites in mammalian genomes. Nucleic Acids Res. 2000;28(21):4364–75.
11. Burset M, Seledtsov IA, Solovyev VV. Splicedb: database of canonical and non-canonical mammalian splice sites. Nucleic Acids Res. 2001;29(1):255–9.

12. Trapnell C, Pachter L, Salzberg SL. Tophat: discovering splice junctions with rna-seq. Bioinformatics. 2009;25(9):1105–11.
13. Au KF, Jiang H, Lin L, Xing Y, Wong WH. Detection of splice junctions from paired-end rna-seq data by splicemap. Nucleic Acids Res. 2010;38(14):4570–8.
14. Wang K, Singh D, Zeng Z, Coleman SJ, Huang Y, Savich GL, He X, Mieczkowski P, Grimm SA, Perou CM, et al. Mapsplice: accurate mapping of rna-seq reads for splice junction discovery. Nucleic Acids Res. 2010;38(18):178.
15. Jaganathan K, Panagiotopoulou SK, McRae JF, Darbandi SF, Knowles D, Li YI, Kosmicki JA, Arbelaez J, Cui W, Schwartz GB, et al. Predicting splicing from primary sequence with deep learning. Cell. 2019;176(3):535–48.
16. Pertea M, Lin X, Salzberg SL. Genesplicer: a new computational method for splice site prediction. Nucleic Acids Res. 2001;29(5):1185–90.
17. Degroeve S, Saeys Y, De Baets B, Rouzé P, Van De Peer Y. Splicemachine: predicting splice sites from high-dimensional local context representations. Bioinformatics. 2004;21(8):1332–8.
18. Sonnenburg S, Schweikert G, Philips P, Behr J, Rätsch G. Accurate splice site prediction using support vector machines, Vol. 8; 2007. p. 7. BioMed Central.
19. Lee T, Yoon S. Boosted Categorical Restricted Boltzmann Machine for Computational Prediction of Splice Junctions. In: International Conference on Machine Learning. France; 2015. p. 2483–92.
20. Lee B, Lee T, Na B, Yoon S. Dna-level splice junction prediction using deep recurrent neural networks. arXiv preprint arXiv:1512.05135. 2015.
21. Zuallaert J, Godin F, Kim M, Soete A, Saeys Y, De Neve W. Splicerover: interpretable convolutional neural networks for improved splice site prediction. Bioinformatics. 2018;34(24):4180–8.
22. Hubbard T, Barker D, Birney E, Cameron G, Chen Y, Clark L, Cox T, Cuff J, Curwen V, Down T, et al. The ensembl genome database project. Nucleic Acids Res. 2002;30(1):38–41.
23. Keren H, Lev-Maor G, Ast G. Alternative splicing and evolution: 614 diversification, exon definition and function. Nat Rev Genet. 2010;11(5):345.
24. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. Lake Tahoe: NIPS (Neural Information Processing Systems Foundation); 2012. p. 1097–105.
25. Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics. Fort Lauderdale: PMLR; 2011. p. 315–23.
26. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res. 2014;15(1):1929–58.
27. Bishop CM. Pattern Recognition and Machine Learning. Cambridge: Springer; 2006.
28. Deng L-Y. The cross-entropy method: A unified approach to combinatorial optimization, monte-carlo simulation, and machine learning. Technometrics. 2006;48(1):147–8.
29. Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. 2014.
30. Umarov R, Kuwahara H, Li Y, Gao X, Solovyev V. Promoter analysis and prediction in the human genome using sequence-based deep learning models. Bioinformatics. 2019;1:8.
31. Shrikumar A, Greenside P, Kundaje A. Learning Important Features Through Propagating Activation Differences. In: Proceedings of the 34th International Conference on Machine Learning. Sydney: ICML; 2017. p. 3145–53. http://proceedings.mlr.press/v70/shrikumar17a.html.
32. Crooks G, Hon G, Chandonia J, Brenner S. WebLogo: a sequence logo generator. Genome Res. 2004;14(6):.
33. Hosmer Jr DW, Lemeshow S, Sturdivant RX. Applied Logistic Regression. vol. 398. Amherst: Wiley; 2013.
34. Breiman L. Classification and Regression Trees. Wadsworth: Routledge; 2017.
35. Breiman L. Random forests. Mach Learn. 2001;45(1):5–32.
36. Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995;20(3):273–97.
37. Hinton GE, Salakhutdinov RR. Reducing the dimensionality of data with neural networks. Science. 2006;313(5786):504–7.
38. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80.
39. Sterck L, Billiau K, Abeel T, Rouze P, Van de Peer Y. Orcae: online resource for community annotation of eukaryotes. Nat Methods. 2012;9(11):1041.
40. Lifton R, Goldberg M, Karp R, Hogness D. The organization of the histone genes in drosophila melanogaster: functional and evolutionary implications. volume 42. In: Cold Spring Harbor Symposia on Quantitative Biology. Cold Spring Harbor: Cold Spring Harbor Laboratory Press; 1978. p. 1047–51.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.