Neural Network Models for Analyzing the Splicing Cell Variable from Genome Sequences

Thesis submitted in partial fulfilment of the requirements for the award of the degree of

Doctor of Philosophy

in

Computer Science and Engineering

by

Aparajita Dutta

Under the supervision of

Dr. Ashish Anand

Department of Computer Science and Engineering
Indian Institute of Technology Guwahati
Guwahati - 781039 Assam India

May, 2021

Dedicated to

My Mother, for being my pillar of strength

and

My Father, for being my eternal guiding light

Acknowledgements

I express my earnest gratitude to my supervisor *Dr. Ashish Anand* for his esteemed guidance and support throughout my work. He has always provided the right balance of guidance and flexibility, which helped me nurture my research interests and abilities. I am indebted to his consistent support, care, time, and wisdom, without which this thesis could not have taken shape. I am grateful to him for encouraging me to learn new skills and for providing me several research exposures through travel support and collaborations.

I express my sincere gratitude to *Dr. Kusum Kumari Singh* (Dept. of BSBE) for her guidance on understanding the complex biological processes. I want to acknowledge the other members of my Doctoral Committee - *Dr. V. Vijaya Saradhi* and *Dr. Arijit Sur* whose timely suggestions and constructive feedback helped me improve my work. I am also thankful to the anonymous reviewers of my research works in various forums for their critical comments, which helped me add quality to my work.

I would also like to acknowledge the heads of the department of Computer Science and Engineering at IITG during my Ph.D. work - *Prof. Diganta Goswami*, *Prof. S. V. Rao* and *Prof. Jatindra Kumar Deka* for their support concerning the departmental resources and facilities. I want to thank the staff of the department - *Nanu Alan Kachari, Bhriguraj Borah, Hemanta Kr. Nath, Nava Kumar Boro, Raktajit Pathak, Pranjit Talukdar, and Prabin Bharali* who were always approachable and welcoming whenever I sought their help. I am also thankful to all the fellow researchers of the department, security persons, and other staff members for their help and support.

I would like to gratefully acknowledge *MHRD*, *Govt. Of India* for their financial support rendered during my Ph.D., without which this research could not have taken shape. I am also grateful to *Dr. Aryabartta Sahu* (Dept. of CSE) for granting access to the GPU on which most of the experiments were carried out. I acknowledge the *Department of Biotechnology, Govt. of India* for the financial support in the project BT/COE/34/SP28408/2018. I also thank my co-authors *Tushar Dubey*, *Aman Dalmia*, and *Athul R*, for their contributions to the work.

I am deeply indebted to the seniors in my lab, *Sunil Kumar Sahu*, *Saptarshi Pyne* and *Abhishek* whose technical guidance helped me sail through the initial phase of the journey. I am also glad to have found friends like *Palash Das* and *Sayan Bhattacharjee* who took care of me like a family at a place away from home. They were my constant support system, and their love, care, and compassion made life easier and beautiful at IIT Guwahati. I would also like to mention friends cum batchmates like *Anasua Mitra* and *Abhijit Das* whose persistent interactions during coursework helped me stay focused. I would also like to acknowledge my

friends at Subansiri hostel of IIT Guwahati - *Niharika*, *Tanushree*, *Protima*, *Saswati*, *Divya* and others who added colours to my hostel life and made this journey a memorable one.

I am also grateful to other friends outside IITG - *Sayanti*, *Tanmaya*, *Pallavi*, *Tina*, *Bhanita*, *Najima*, *Amrita* and many more who did not lose touch with me and always extended their love, support and encouragement through all these years. I have, on several occasions, confided in them with issues that affected me at a personal or professional level, and they have always patiently listened and understood my situation without any judgments.

This journey would not have been possible without the constant support, unconditional love, and profound encouragement of my mother *Smt. Tapati Dutta*. Her life, to me, is a living example of courage, persistence, and perseverance. Although I could not have the physical presence of my father *Lt. Dr. Shyamal Kanti Dutta* during my journey of Ph.D., I have always felt the presence of his blessings and guidance. I also want to extend my heartfelt gratitude and appreciation towards my spouse *Dr. Pavel Sikidar* for never doubting my capabilities and for whole-heartedly supporting me in all my endeavors. I also fall short of words to express my gratitude to my extended family for understanding the importance of this journey in my life and for never questioning my priorities.

May 06, 2021 Aparajita Dutta

Declaration

I certify that

- The work contained in this thesis is original and has been done by myself and under the general supervision of my supervisor(s).
- The work reported herein has not been submitted to any other Institute for any degree or diploma.
- Whenever I have used materials (concepts, ideas, text, expressions, data, graphs, diagrams, theoretical analysis, results, etc.) from other sources, I have given due credit by citing them in the text of the thesis and giving their details in the references. Elaborate sentences used verbatim from published work have been clearly identified and quoted.
- I also affirm that no part of this thesis can be considered plagiarism to the best of my knowledge and understanding and take complete responsibility if any complaint arises.
- I am fully aware that my thesis supervisor(s) are not in a position to check for any possible instance of plagiarism within this submitted work.

May 06, 2021 Aparajita Dutta

Department of Computer Science and Engineering
Indian Institute of Technology Guwahati
Guwahati - 781039 Assam India

Dr. Ashish Anand
Associate Professor
Email : anand.ashish@iitg.ernet.in
Phone : +91-361-2582374

Certificate

This is to certify that this thesis entitled "**Neural Network Models for Analyzing the Splicing Cell Variable from Genome Sequences**" submitted by **Aparajita Dutta**, in partial fulfilment of the requirements for the award of the degree of Doctor of Philosophy, to the Indian Institute of Technology Guwahati, Assam, India, is a record of the bonafide research work carried out by him under my guidance and supervision at the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati, Assam, India. To the best of my knowledge, no part of the work reported in this thesis has been presented for the award of any degree at any other institution.

Date: May 06, 2021
Place: Guwahati

Dr. Ashish Anand
(Thesis Supervisor)

Abstract

The escalating rate of deaths caused by complex human diseases has led to the need for unraveling the underlying genetic causation of these diseases. The study of the functional and structural information encoded in the genome facilitates genome annotation and helps in deciphering the relationship between the genome (genotype) and the disease traits (phenotype). The relationship between genotype and phenotype is critically complex, passing through several layers of complex biophysical processes. Variations in biophysical processes are manifested through the change in rate and quantity of production of several cell variables like splicing, transcription rate, polyadenylation, and DNA methylation. It is easier to associate the genotype with such more closely related measurable intermediate cell variables.

This thesis focuses on studying one such cell variable called splicing. Splicing occurs co-transcriptionally during RNA processing of genes. A gene comprises alternating regions called exons and introns. During splicing, the introns of a gene are removed, and the exons are ligated. Splicing is responsible for the transcript and protein diversity in eukaryotes. Several computational models are employed for gaining a deeper understanding of the splicing phenomenon. One way of studying the splicing mechanism is to identify splice sites from genome sequences by employing computational models.

However, the existing studies on the identification of splice sites have one or more of the following limitations:

1. The traditional computational models that identify splice sites are mainly based on functional genomic features. However, such feature sets are neither exhaustive nor optimal.

2. Several existing studies do not focus on extraction and interpretation of the biological features learnt by the model.

3. The existing studies primarily focus on identifying canonical splice sites: sites that contain the consensus GT and AG at donor and acceptor sites.

4. Most of the existing studies focus on studying splice sites from a single species.

This thesis works on the limitations mentioned above. We aim at identifying splice sites based on sequence-based features only. We employ various neural network models that learn the sequence-based features by themselves such that hand-crafted feature engineering can be eliminated to a great extent. This reduces the dependency on the existing knowledge bias.

Neural network models have obtained state-of-the-art performances in identifying splice sites from the genome sequences de novo. Often such models take only nucleotide sequences as input and learn relevant features on their own. However, extracting the interpretable motifs from the model remains a challenge. We explore several existing visualization techniques in their ability to infer relevant features learnt by a neural network on the task of splice junction identification. We study a particular class of neural networks, called recurrent neural networks (RNN), in this thesis.

The existing prediction models primarily focus on identifying canonical splice sites. However, identification of non-canonical splice sites (splice sites lacking the $GT - AG$ consensus) is also equally important for a comprehensive understanding of the splicing phenomenon. This thesis works towards this objective by studying non-canonical splice sites in greater detail to obtain features specific to the non-canonical splicing.

Furthermore, most of the existing studies focus on identifying and analyzing splice sites for a single species. However, models capable of identifying splice sites from multiple species with comparable accuracy are preferable due to the robustness and generalizability. We analyze the performance of an RNN model in identifying splice sites from human, mouse, and drosophila melanogaster. We also test the model's performance on species that were not used during training. Furthermore, we extract the non-canonical splicing features learnt by the model from the three species and validate them with knowledge from the literature.

Contents

Abstract xi

List of Figures xvii

List of Figures xx

List of Tables xxi

List of Tables xxii

List of Abbreviations xxiii

1 Introduction 1
1.1 Objectives of the Thesis . . . 2
1.2 Contributions of the thesis . . . 3
1.3 Outline of the Thesis . . . 9

2 Biological Background 11
2.1 Why Should we study the genome? . . . 11
2.2 Cell variables and their association with diseases . . . 13
2.3 Understanding Genome . . . 14
2.4 Understanding Gene Expression . . . 14
2.5 Splicing regulation . . . 16
2.6 Chapter Summary . . . 16

3 Literature Survey 19
3.1 Challenges in identification of splice sites . . . 19
3.2 Models for splice site identification . . . 20
3.2.1 Sequence alignment-based and statistical techniques . . . 20
3.2.2 Machine learning based techniques . . . 21
3.3 Features for splice sites identification . . . 22
3.3.1 Hand engineered feature set . . . 22
3.3.2 Optimized feature set . . . 23
3.3.3 Self-learnt feature set . . . 24
3.4 Limitations of the current literature . . . 24
3.4.1 Extraction of splicing features . . . 24
3.4.2 Identification of non-canonical splice sites . . . 25
3.4.3 Identification of splice sites in multiple species . . . 26

3.5 Chapter Summary . . . 26

4 SpliceVec: distributed feature representations for splice junction prediction 29
4.1 Introduction . . . 29
4.2 Methods . . . 30
4.2.1 Data . . . 31
4.2.2 Distributed representation . . . 31
4.2.3 Distributed representation of splice junctions . . . 32
4.2.3.1 Genome based SpliceVec . . . 32
4.2.3.2 Splicing-context based SpliceVec . . . 32
4.2.4 SpliceVec feature space construction . . . 33
4.2.5 Classification of SpliceVec by MLP . . . 34
4.3 Results . . . 35
4.3.1 Qualitative analysis of SpliceVec . . . 35
4.3.2 Optimal sequence length for SpliceVec . . . 35
4.3.3 Improved classification of splice junctions . . . 37
4.3.4 Robust classification of SpliceVec-MLP . . . 38
4.3.5 Prediction performance . . . 39
4.4 Chapter Summary . . . 41

5 SpliceVisuL: Visualization of Bidirectional Long Short-term Memory Networks for Splice Junction Prediction 43
5.1 Introduction . . . 43
5.2 Methods . . . 45
5.2.1 Preliminaries on model architecture . . . 45
5.2.1.1 Recurrent neural networks (RNN) . . . 45
5.2.1.2 Long short-term memory (LSTM) units . . . 46
5.2.1.3 Bidirectional long short-term memory (BLSTM) networks . . . 46
5.2.1.4 Attention mechanism . . . 47
5.2.2 Neural architecture . . . 48
5.2.2.1 Input representation . . . 48
5.2.2.2 Modeling splice junctions using BLSTM network . . . 48
5.2.2.3 Feature interpretation using attention layer . . . 49
5.2.3 Visualization techniques . . . 49
5.2.3.1 Smooth gradients of noisy nucleotide embeddings . . . 50
5.2.3.2 Integrated gradients of nucleotide embeddings . . . 50
5.2.3.3 Omission of a single nucleotide . . . 51
5.2.3.4 Occlusion of k-mers . . . 51
5.2.4 Prerequisites for visualization . . . 52
5.3 Experimental setup . . . 53
5.3.1 Positive data generation . . . 53
5.3.2 Negative data generation . . . 54
5.3.3 Training and Hyperparameter tuning . . . 55
5.3.4 Variation in length of flanking region . . . 56

5.3.5 Variation in model architecture . . . 56
5.3.6 Baselines . . . 57
5.3.7 Input formation for SpliceVec-MLP . . . 57
5.3.8 Hyperparameters of the various baselines . . . 58
5.4 Results . . . 58
5.4.1 Prediction performance . . . 58
5.4.2 Visualization of splicing features . . . 59
5.4.2.1 Identifying the significance of sequence positions near splice junctions . . . 59
5.4.2.2 Identifying the splicing motifs at splice junctions . . . 60
5.4.2.3 Identifying the location of branchpoint consensus 'CTRAY' 62
5.4.2.4 Identifying the optimal motif length per position . . . 66
5.5 Discussion . . . 67
5.6 Chapter Summary . . . 68

6 SpliceViNCI: Visualizing the splicing of non-canonical introns through recurrent neural networks 69
6.1 Introduction . . . 69
6.2 Methods . . . 70
6.2.1 Neural architecture . . . 71
6.2.2 Visualization techniques . . . 72
6.3 Experimental Setup . . . 72
6.3.1 Dataset . . . 72
6.3.1.1 Positive data . . . 72
6.3.1.2 Negative data . . . 72
Randomness-based negative data: . . . 72
Consensus-based negative data: . . . 73
6.3.2 Training and hyperparameter tuning . . . 74
6.3.3 Baselines . . . 75
6.4 Results . . . 76
6.4.1 SpliceViNCI learns better representations of non-canonical splice junctions . . . 76
6.4.2 Non-canonical splicing features are relatively further from the splice junctions . . . 76
6.4.3 SpliceViNCI outperforms state-of-the-art splice junction prediction models . . . 78
6.4.4 Donor and acceptor splicing signals identify the splice junctions cooperatively . . . 79
6.4.5 Identification of novel non-canonical splice junctions by SpliceViNCI . 80
6.4.6 Visualization of splicing features captured by SpliceViNCI . . . 81
6.4.6.1 Significance of sequence positions . . . 81
6.4.6.2 Optimal feature length per position . . . 83
6.4.6.3 Importance of each nucleotide per position . . . 83
6.4.6.4 Most important motifs in a specific region . . . 84
6.5 Chapter Summary . . . 85

7 SpliceTrans: Transferring knowledge across species for identification of splice junctions **87**
7.1 Introduction . . . 87
7.2 Methods . . . 89
7.3 Experimental Setup . . . 89
7.3.1 Dataset . . . 89
7.3.1.1 Positive data . . . 89
7.3.1.2 Negative data . . . 90
7.3.2 Training and hyperparameter tuning . . . 90
7.3.3 Baseline . . . 91
7.4 Results . . . 93
7.4.1 SpliceTrans outperforms state-of-the-art in identifying splice sites of multiple species . . . 93
7.4.2 SpliceTrans outperforms state-of-the-art in identifying splice sites of unseen species . . . 93
7.4.3 SpliceTrans is more robust with imbalanced training data . . . 95
7.4.4 SpliceTrans outperforms state-of-the-art in identifying splice sites of partially annotated species . . . 95
7.4.5 SpliceTrans captures significant sequence positions . . . 100
7.4.5.1 Significant sequence positions captured in mouse . . . 100
7.4.5.2 Significant sequence positions captured in drosophila . . . 101
7.4.6 SpliceTrans captures splice junction consensus . . . 102
7.5 Chapter Summary . . . 102

8 Conclusion and Future Directions **105**
8.1 Conclusions . . . 105
8.2 Future Directions . . . 107

Bibliography **109**

Publications **121**

Vitae **123**

List of Figures

1.1 Three layer association of genotype, phenotype and cell variables. 2

1.2 **t-SNE plots for different embeddings.** (a) Random embedding (b) SpliceVec-g (c) SpliceVec-sp. Each point represents a splice junction. Points in red represent false splice junctions whereas points in blue represent true splice junctions. 4

1.3 Thesis overview 8

2.1 A portion of chromosome 5 that contains the two SMN genes 12

2.2 An overview of the important stages involved in the process of gene expression. 15

2.3 Existing consensus in the canonical (a) donor and (b) acceptor splice junctions of the dataset. The consensus comprises the extended donor site consensus 9-mer $[AC]AGGTRAGT$ and the extended acceptor site consensus 15-mer $Y_{10}NCAGG$. The zero, negative, and positive indices represent the splice junction and its upstream and downstream regions. 16

4.1 **Proposed approach:** (a) feature representation; (b) splice junction classification. 30

4.2 **Architecture of word2vec and doc2vec models.** (a) CBOW for word2vec (b) skip-gram for word2vec (c) DM for doc2vec and (d) DBOW for doc2vec. 33

4.3 **t-SNE plots for different embeddings.** (a) Random embedding (b) SpliceVec-g (c) SpliceVec-sp. Each point represents a splice junction. Points in red represent false splice junctions whereas points in blue represent true splice junctions. 36

4.4 **Performance of SpliceVec with increasing ratio of negative samples to positive samples.** (a) Precision (b) Recall (c) F1-score of SpliceVec-sp, SpliceVec-g and DeepSplice on varying ratio of negative samples from 5 to 17. Data here is positive and negative samples from chromosome 20 of GENCODE dataset. 39

5.1 An unrolled RNN. 45

5.2 An LSTM cell. 47

5.3 A BLSTM network. 48

5.4 An overview of SpliceVisuL architecture. 49

5.5 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. 55
5.6 t-SNE plots of 4 random sets of 1000 true and 1000 decoy splice junctions. Points in blue represent true, whereas points in yellow represent decoy splice junction pairs. 59
5.7 **Significance of sequence positions near splice junctions.** The average deviation value per nucleotide position is shown for occlusion-1 of canonical (a) donor and (b) acceptor splice junctions, smooth gradient of canonical (c) donor and (d) acceptor splice junctions, (e) occlusion-1 of non-canonical acceptor and (f) smooth gradient of non-canonical acceptor splice junctions. 61
5.8 **The splicing motifs at canonical splice junctions.** The average deviation value per position per nucleotide is shown for canonical donor junction motifs obtained from (a) occlusion-1 (b) occlusion-3 (c) omission (d) integrated gradients and canonical acceptor junction motifs obtained from (e) occlusion-1 (f) occlusion-3 (g) omission and (h) smooth gradients. 63
5.9 Average deviation value per position per nucleotide based on attention weights of canonical sequences. 64
5.10 **The splicing motifs at non-canonical splice junctions.** The average deviation value per position per nucleotide is shown for non-canonical donor junction motifs obtained from (a) smooth gradients and (b) integrated gradients as well as non-canonical acceptor junction motifs obtained from (c) occlusion-3 and (d) smooth gradients. 64
5.11 **The average deviation value of branchpoint pattern CTRAY in the upstream region [-40, -15] of acceptor splice junction.** The average deviation value per position for the pattern CTRAY is shown for (a) attention (b) smooth gradients (c) integrated gradients (d) omission (e) occlusion-1 (f) occlusion-3 of canonical acceptor splice junctions (g) integrated gradients and (h) occlusion-3 of non-canonical acceptor splice junctions. 65
5.12 **The optimal motif length per position.** The frequency of different window lengths, varying from 1 to 11 (represented by E), is shown for occlusion of canonical (a) donor and (c) acceptor splice junctions and occlusion of non-canonical (b) donor and (d) acceptor splice junctions. 66

6.1 A schematic of the network architecture. 71
6.2 **t-SNE plots of non-canonical splice junctions obtained from SpliceViNCI for (a) Type-1 dataset, (b) Type-2 dataset and from SpliceRover for (c) Type-1 dataset, (d) Type-2 dataset.** The points in blue are positive splice junctions whereas points in red are negative splice junctions. 77
6.3 **Performance of various state-of-the-art models.** The performance is measured in terms of F1-score for canonical (*can*) and non-canonical (*non-can*) splice junctions from both Type-1 and Type-2 dataset. F1-score is computed in percentage. 79

6.4 **Performance of SpliceViNCI on *seen* and *unseen* data.** (a) The number of *seen* and *unseen* dimer pairs identified by SpliceViNCI from Type-1 and Type-2 dataset. (b) The number of *seen* and *unseen* dimer pairs identified by SpliceViNCI and all the baselines from Type-2 dataset. 81

6.5 **The significance of sequence position.** The average deviation value per position is shown for non-canonical donor junctions by (a) integrated gradients (b) occlusion-1, and non-canonical acceptor junctions by (c) integrated gradients (d) occlusion-1. 82

6.6 **The optimal motif length per position.** The frequency of different window lengths, varying from 1 to 11, is shown for occlusion of non-canonical (a) donor and (b) acceptor splice junctions. 83

6.7 **The average deviation value per position per nucleotide.** The average deviation value per position per nucleotide is shown for non-canonical donor junctions by (a) integrated gradients (b) occlusion-1 and non-canonical acceptor junctions by (c) integrated gradients (d) occlusion-1. 84

6.8 The frequency of various k-mers, given by occlusion, in the region -30 nt to -80 nt upstream of the non-canonical acceptor junction. 85

7.1 Performance (in percentage) obtained by SpliceTrans, SpliceRover and SpliceFinder in the identification of canonical (can) and non-canonical (non-can) splice junctions when trained and tested with data from the same species. 92

7.2 Performance (in percentage) obtained by SpliceTrans, SpliceRover and SpliceFinder trained with human data in the identification of canonical (can) and non-canonical (non-can) splice junctions from mouse and drosophila test data. . . 94

7.3 F1-score (in percentage) obtained by SpliceTrans, SpliceFinder and SpliceRover in the identification of splice junctions with imbalanced training data. 96

7.4 F1-score (in percentage) obtained by SpliceTrans, SpliceFinder and SpliceRover in the identification of splice junctions with single species and multi-species training data. 97

7.5 **The importance of sequence positions.** The average deviation value per position obtained by integrated gradients for non-canonical (a) donor junction and (b) acceptor junction in mouse+human model; non-canonical (c) donor junction and (d) acceptor junction in mouse model. 98

7.6 **The importance of sequence positions.** The average deviation value per position obtained by integrated gradients for non-canonical (a) donor junction and (b) acceptor junction in drosophila+human model; non-canonical (c) donor junction and (d) acceptor junction in drosophila model. 99

7.7 **Sequence motifs at splice junctions.** The average deviation value per position per nucleotide is shown by integrated gradients for non-canonical (a) donor junctions and (b) acceptor junctions in mouse+human model; non-canonical (c) donor junctions and (d) acceptor junctions in mouse model. . . 100

7.8 **The importance of nucleotides per sequence position.** The average deviation value per nucleotide per position is shown by integrated gradients for non-canonical (a) donor junction and (b) acceptor junction in drosophila+human model; non-canonical (c) donor junction and (d) acceptor junction in drosophila model. 101

List of Tables

1.1 Performance of splice junction classification using SpliceVec-g and SpliceVec-sp by varying length of junction sequences. We compute accuracy (Ac), precision (Pr), recall (Re), and F1 score (F1) in percentage as performance measures. The values are average of five simulations. 5
1.2 Summary of the various visualization techniques in their ability to identify selected canonical splicing features. 6
1.3 F1-score (in percentage) obtained by SpliceViNCI in identification of canonical (can) and non-canonical ($non - can$) splice junctions with varying flanking region on Type-1 and Type-2 dataset. 7

2.1 Cell variables related to genomic regulatory mechanisms [Adapted from [76]] 13

3.1 Machine learning approaches for predicting the splicing regulations 22

4.1 Optimal hyper-parameters for word2vec and doc2vec training models. 34
4.2 Optimal hyper-parameters of the classifiers. 35
4.3 Performance of splice junction classification using SpliceVec-g and SpliceVec-sp by varying length of junction sequences. We compute accuracy (Ac), precision (Pr), recall (Re) and F1 score (F1) in percentage as performance measures. The values are average of five simulations. 37
4.4 Performance comparison of different models for splice junction prediction. Performance of SpliceVec is obtained by considering optimal sequence length of 10nt flanking region including complete intronic sequence. 37
4.5 Performance comparison of SpliceVec-MLP with SpliceMachine for prediction of donor and acceptor sites individually. 38
4.6 Classification by reducing training dataset: Performance comparison of DeepSplice and SpliceVec. 39
4.7 Correctly predicted canonical ($Pred_{can}$) and non-canonical ($Pred_{non-can}$) positive splice junctions, out of 87,927 canonical and 226 non-canonical junctions, using MLP and both SpliceVec-g and SpliceVec-sp by varying the length of junction sequence. 40

5.1 Performance of the various models in donor splice junction and junction pair prediction. Accuracy (Ac), Precision (Pr), Recall (Re) and F1 Score (F1) are computed in percentage. 54
5.2 Performance of the model with variation in flanking region 56
5.3 Performance of the model with variation in architecture 57

5.4 Performance of *SpliceVisuL* compared with state-of-the-art models. Accuracy (Ac), Precision (Pr), Recall (Re), and F1 Score (F1) are computed in percentage. 60
5.5 Summary of the various visualization techniques in their ability to identify selected canonical splicing features. 67

6.1 Distribution of canonical and non-canonical splice junctions in the positive and negative data. 74
6.2 Distribution of the top 2 most frequent non-canonical dimer pairs in the positive and negative data. 74
6.3 F1-score (in percentage) obtained by SpliceViNCI in identification of canonical (*can*) and non-canonical (*non-can*) splice junctions with varying flanking region on Type-1 and Type-2 dataset. 78
6.4 Performance of various state-of-the-art models on Type-2 dataset considering donor, acceptor and donor-acceptor junction pair as input. F1-score (in percentage) is computed as the performance metric. 80

7.1 Distribution of positive dataset in mouse, human and Drosophila melanogaster. 90

List of Abbreviations

Terms	Abbreviations
DNA	deoxyribonucleic acid
RNA	Ribonucleic acid
snRNA	small nuclear RNA
MLP	Multilayer perceptron
BLSTM	Bidirectional Long Short-Term Memory
RNN	Recurrent Neural Network
SMA	Spinal Muscular Atrophy
SMN	Survival Motor Neuron
MDA	Muscular Dystrophy Association
mRNA	messenger-RNA
MISO	mixture-of-isoforms
EMG	expression minigenes
HMM	Hidden Markov Model
SVM	Support Vector Machine
CNN	Convolutional Neural Network
nt	nucleotide
NLP	Natural Language Processing

CBOW	Continuous Bag Of Words
DBOW	Distributed Bag Of Words
DM	Distributed Memory
LSTM	Long Short-Term Memory
GRU	Gated Recurrent Unit
ESTs	Expressed Sequence Tags
t-SNE	t-distributed Stochastic Neighbor Embedding
PY-tract	polypyrimidine tract
AS	Alternative Splicing
LRP	Layer-wise Relevance Propagation
RF	Random Forest
DT	Decision Trees
NB	Naive Bayes

"You have to dream before your dreams can come true."

A.P.J. Abdul Kalam (1931 - 2015)
Indian scientist and leader

1

Introduction

We are in an era where death caused due to complex human diseases like cancer, diabetes, and autism are increasing at alarming rates [108]. To design a better cure for such diseases, we need to understand the underlying causes of the disease. The study of the functional and structural information encoded in the genome facilitates genome annotation and helps in deciphering the relationship between the genome and the disease traits.

One straightforward approach can be to develop computational tools to predict the relationship between the genome and the physical traits and disease risks. Input to such computational models can be the genome sequence, also called the genotype. The output of the model can be the physical traits, also called the phenotype. However, the direct association between genotype and phenotype is not ideal. The genotype-phenotype relationship is critically complex, passing through several layers of intricate and interrelated biophysical processes shaped through generations of evolution.

Another approach is to associate the two ends via a layer of measurable intermediate state called molecular phenotypes or cell variables (Figure 1.1) [76]. Splicing patterns, polyadenylation, locations where a protein binds to a strand of DNA containing a gene, the number of copies of a gene in a cell, and protein concentration can be a few examples of cell variables. The association between genotype and cell variable is more closely related than the association between genotype and phenotype. Therefore, the genotype-cell variable association can be easily comprehended by learning models. These measurable intermediate cell variables correspond to biochemically active components whose production quantities can be directly modulated by disease risks. Hence, developing an understanding of the cell variables can provide us greater insights into the genome's structural and functional properties and eventually help us correlate the disease causation to the genome functions.

We aim to explore a crucial cell variable involved in the RNA processing, called splicing, which contributes to the transcript and protein diversity in eukaryotes [116]. Splicing removes certain regions of the gene called *introns* and ligates the regions called *exons*. However, often the exons and introns are alternatively joined or skipped leading to the formation of different transcripts and distinct protein isoforms. This phenomenon is called *alternative*

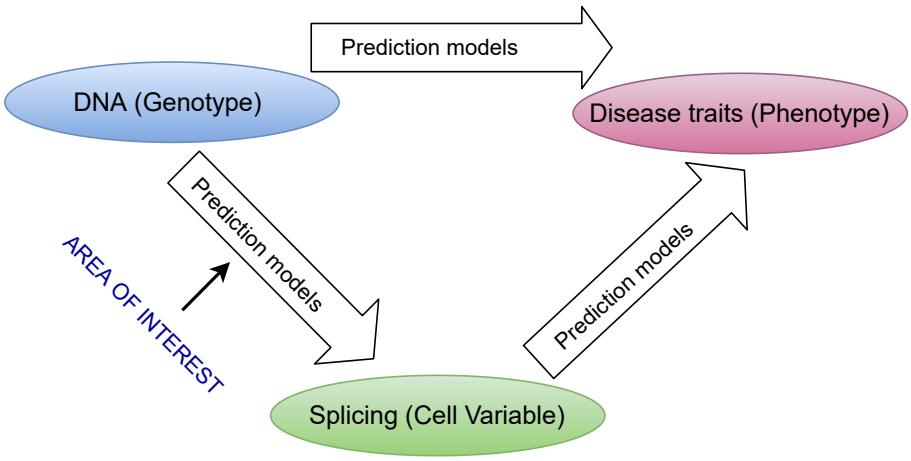


Figure 1.1: Three layer association of genotype, phenotype and cell variables.

splicing. Splicing occurs at the junctions between exons and introns called *splice sites* or *splice junctions*. The exon-intron junction is called the *donor* site, whereas the intron-exon junction is called the *acceptor* site.

López-Bigas et al. observed that up to 60% of genetic disorders caused by genomic mutations are related to alterations in the splicing process [82]. Some of the major splicing-related diseases include neurological and psychiatric disorders, spinal muscular atrophy, Parkinsonism, cancers, and haemophilia [27, 66, 85, 123]. Therefore, the benefit of developing a deeper understanding of the splicing mechanism is necessary. It helps us unravel the genomic structure and functions. It also helps us in understanding the relationship between splicing regulation and disease traits.

The inclusion and exclusion of introns and exons are catalyzed by a large RNA-protein complex called spliceosome. The spliceosome is an assembly of small nuclear RNAs (snRNAs) and numerous other proteins. This protein assembly selectively binds to the genome sequence based on sequence patterns that act as regulatory instructions or signals guiding the protein-sequence interactions. Therefore, identification of these regulatory instructions are required to identify the splice sites accurately.

1.1 Objectives of the Thesis

Presently, the vast availability of annotated sequences makes it possible to create large enough training datasets for supervised learning algorithms. Therefore, several machine learning methods, including deep learning methods, have been applied to identify cell variables like transcription factor binding [67], alternative splicing [14], and polyadenylation [61]. Our interest lies in understanding the splicing mechanism through the application of computational models. However, most of the existing studies which employ computational models to identify splice sites have several limitations. We discuss some of the major limitations

which form the objectives of our research.

1. The traditional computational models are based on manual extraction and selection of functional genomic features for training the learning model. Such manually engineered features are not optimal or exhaustive. Hence, they affect the performance of the computational model. This has motivated research based on feature selection techniques that can identify the optimal set of features relevant to splice site identification. However, such a feature set is still biased by the existing knowledge.

2. To remove the dependency on hand-crafted features, researchers were motivated to adopt models that can identify sequence-based features specific to cell variables without any manual extraction and selection of features. Instead, the model itself can capture features from the genome sequences that act as regulatory signals. Researchers have applied this approach for the prediction of cell variables like transcription factor binding and splicing. However, the idea is still nascent and different models and visualization techniques still need to be explored to compare and identify the more suited method for such a task.

3. Although there has been work done to explore important regulatory features in the domain of splicing, we still believe that not all of the features are known yet, specifically in the case of non-canonical splicing [75]. Most of the existing computational models identify canonical splice sites only. However, identification of non-canonical splice sites is also equally crucial for a comprehensive understanding of the splicing mechanism [75].

4. Furthermore, most of the existing methodologies identify splice sites from a single species. However, it is desirable to explore the performance of a predictive model in identifying splice sites from multiple species [35]. Such a model can be considered more robust and generalizable.

1.2 Contributions of the thesis

We propose various neural network models to primarily identify splice sites and extract the splicing features from genome sequences. The input to all the neural network models is genome sequences. However, the types of genome sequences fed into the model vary with the objectives. The input needs to be converted to vector form before feeding into the neural network. Since genome sequences are a continuous stretch of nucleotides, we characterize the sequence representations at different levels: character, word, and sentence.

The contributions of this thesis can be broadly divided into the stages described below. The contributions are corresponding to the objectives described in Section 1.1.

Contribution 1: In the first stage, we propose a model, named SpliceVec-MLP, which identifies canonical and non-canonical splice sites by learning features *de novo* from

genome sequences. In this process, the model is not fed any manually extracted features. Hence, the model can also take into account unknown features that may be regulating the splicing phenomenon. Here, the need for feature engineering has been eliminated to a great extent. Here, we explore genome sequence representation in the form of words and sentences. The key findings of this contribution are summarized below:

1. We propose two variations of SpliceVec (SpliceVec-g and SpliceVec-sp) for feature representation of splice sites at word and sentence levels, respectively. The feature representations are classified as true or decoy splice sites using a multilayer perceptron (MLP) as the classifier. SpliceVec-MLP outperforms state-of-the-art methods by 2.42-18.86% in terms of accuracy for splice site prediction.
2. An intrinsic evaluation of SpliceVec indicates that it can group true and false sites distinctly as seen in the t-SNE plot in Figure 1.2.

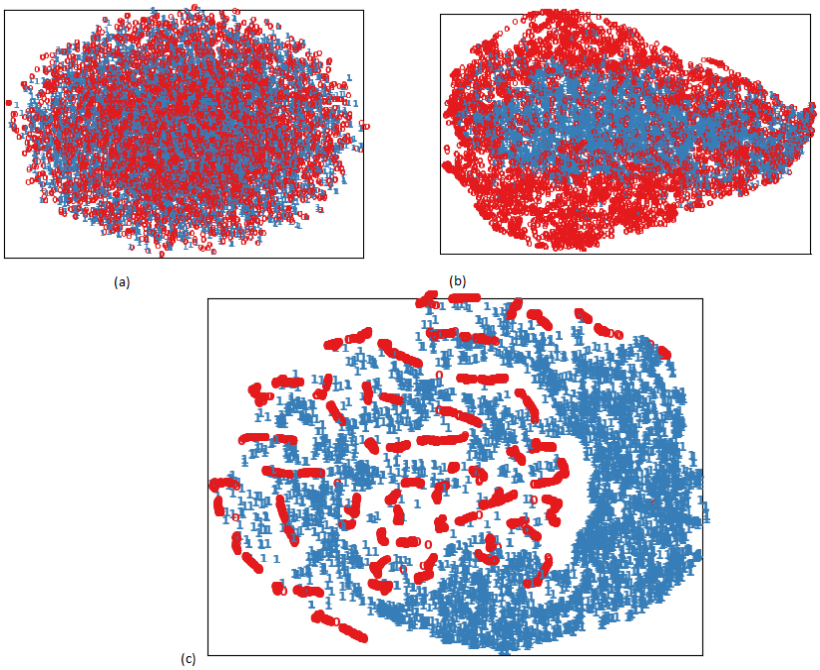


Figure 1.2: t-SNE plots for different embeddings. (a) Random embedding (b) SpliceVec-g (c) SpliceVec-sp. Each point represents a splice junction. Points in red represent false splice junctions whereas points in blue represent true splice junctions.

3. We explored the optimal sequence length that best captures the splicing signals for improving the prediction results. We find that the inclusion of the entire intronic sequence significantly boosts the predictive power of the classifier as observed in Table 1.1.
4. SpliceVec-MLP identifies non-canonical splice junctions with 100% accuracy, indicating that our feature representations are invariant to both canonical and non-canonical splice junctions.

Table 1.1: Performance of splice junction classification using SpliceVec-g and SpliceVec-sp by varying length of junction sequences. We compute accuracy (Ac), precision (Pr), recall (Re), and F1 score (F1) in percentage as performance measures. The values are average of five simulations.

Sequence length	SpliceVec-g	SpliceVec-sp
	$Ac, Pr, Re, F1(\%)$	$Ac, Pr, Re, F1(\%)$
10nt flanking	82.26, 80.61, 85.02, 82.73	99.77, 99.80, 99.73, 99.77
20nt flanking	81.09, 78.85, 85.01, 81.79	99.55, 99.50, 99.61, 99.55
30nt flanking	82.68, 80.67, 85.98, 83.22	99.52, 99.45, 99.58, 99.52
40nt flanking	84.81, 82.87, 87.80, 85.24	99.37, 99.33, 99.41, 99.37
10nt flanking + intron	**98.15, 98.40, 97.89, 98.14**	99.88, 99.81, 99.95, 99.88
20nt flanking + intron	97.73, 97.97, 97.48, 97.72	99.94, 99.92, 99.96, 99.94
30nt flanking + intron	97.35, 97.86, 96.82, 97.34	99.93, 99.92, 99.93, 99.93
40nt flanking + intron	97.04, 97.30, 96.77, 97.03	**99.97, 99.97, 99.97, 99.97**

5. The proposed feature representations are more robust in handling reduced training samples. SpliceVec maintains an accuracy above 99% even with a 60% reduction of training samples, whereas the accuracy of its counterpart drops by about 6%.
6. SpliceVec is more consistent in its performance with class-imbalanced data making it more suitable for the actual scenario where the number of pseudo sites is several times more than that of true splice sites.
7. SpliceVec-MLP, being 12.94 times computationally faster than the state-of-the-art model, contributes as a suitable option for classification of the abundant annotated sequences available these days by high-throughput sequencing technologies. SpliceVec can be trained with user-defined data as well.

Contribution 2: Although SpliceVec shows promising performance in identifying both canonical and non-canonical splice sites, it is not possible to extract the splicing features learnt by the model. This limitation is due to the inability to intuitively explain the N-dimensional embedding space representing the genome sequence in word and sentence format.

Hence, in the second stage, we propose a BLSTM model with attention mechanism named SpliceVisuL, which identifies canonical and non-canonical splice sites from genome sequences. Furthermore, we extract the features modulating the splicing mechanism by applying some widely used visualization techniques. The genome sequences are converted to vector representations at character level before feeding into SpliceVisuL. The key findings of this contribution can be summarized as follows:

1. We generate two different types of the negative dataset to test the consistency of

RNN models compared to other neural network models. The proposed architecture achieves state-of-the-art performance on both types of datasets.

2. We redesign some of the effective visualization techniques, available in the literature, to comprehend genome sequences as inputs. We categorize the visualization techniques into two broad categories: perturbation based and back-propagation based.
3. Results indicate that the visualization techniques produce a comparable performance for branchpoint detection. In the case of canonical donor and acceptor junction motifs, perturbation based visualizations perform better than back-propagation based visualizations and vice-versa for non-canonical motifs.
4. We infer relevant biological information learnt by the model for both canonical and non-canonical splicing events. The splicing features are validated with the existing knowledge from the literature.
5. We further compare and discuss the ability of the visualization techniques in the inference of various known features as shown in Table 1.2.

Table 1.2: Summary of the various visualization techniques in their ability to identify selected canonical splicing features.

Features	Attention	Occlusion	Omission	Smooth gradients	Integrated gradients
Importance of sequence position	✓	✓	✓	✓	✓
Donor site consensus	✗	✓	✓	✗	✓
Acceptor site consensus	✗	✓	✓	✓	✗
Branchpoint	✓	✓	✓	✓	✓

Contribution 3: There is evidence in the literature suggesting that the signals regulating canonical and non-canonical splicing are possibly different from one another [95]. Therefore, in the third stage, we mainly focus on studying non-canonical splice sites and the features governing their regulation. We seek to attain optimal performance for the identification of non-canonical splicing in particular.

We present a BLSTM model named SpliceViNCI for the identification of splice sites. SpliceViNCI is similar to SpliceVisuL, except that the attention layer is removed in this model. We remove the attention layer from SpliceViNCI since this visualization mechanism failed to extract most of the splicing features, as observed in Table 1.2.

The key findings of this work are summarized below.

1. We design the Type-1 and Type-2 datasets based on two different types of negative data and analyze the performance of various state-of-the-art models and the proposed model with both the dataset. SpliceViNCI attains state-of-the-art performance for the identification of both canonical and non-canonical splice junctions.

2. We analyze the length of flanking region required for obtaining the optimal performance in identifying non-canonical splice junctions (Table 1.3). We obtain the optimal performance for the prediction of canonical splice junctions at 60 to 80 nt context. On the contrary, we obtain the optimal performance for the prediction of non-canonical splice junctions at a flanking region of 120 nt.

Table 1.3: F1-score (in percentage) obtained by SpliceViNCI in identification of canonical (can) and non-canonical ($non-can$) splice junctions with varying flanking region on Type-1 and Type-2 dataset.

Flanking region	Type-1 dataset		Type-2 dataset	
	can	non-can	can	non-can
180	99.50	69.71	99.13	97.38
160	99.39	65.47	99.06	97.24
140	99.65	72.09	99.05	97.66
120	99.65	**74.04**	99.05	**97.67**
100	99.67	73.56	99.04	96.82
80	**99.70**	70.31	99.02	96.81
60	99.65	71.06	**99.07**	96.40
40	99.60	69.32	98.30	95.65
20	98.60	60.01	95.82	93.65

3. SpliceViNCI outperforms state-of-the-art models in the identification of novel splice junctions. Previously, only a few machine learning based approaches worked on the identification of novel splice junctions [37, 38, 138, 139]. However, those methods either identified only canonical splice junctions [138, 139] or had limited visualization capabilities of non-canonical splicing features [37, 38].
4. We apply two effective visualization techniques to discern the non-canonical splicing features learnt by the model: integrated gradients and occlusion. The findings thereof are validated with the existing knowledge from the literature. Integrated gradient extracts features that comprise contiguous nucleotides, whereas occlusion extracts features that are individual nucleotides distributed across the sequence.

Contribution 4: The previous three contributions identified splice sites from the human genome only. In the fourth and final stage, we explore the model's performance, proposed in the previous contribution, for identifying splice sites from multiple species. We name this model as SpliceTrans. This contribution tests the generalizability and robustness of SpliceTrans. We also explore how the model performs when tested with a species on which it was not trained. Furthermore, we extract splicing features of different species learnt by the model and validate them with existing knowledge.

The key findings of this work can be summarized as:

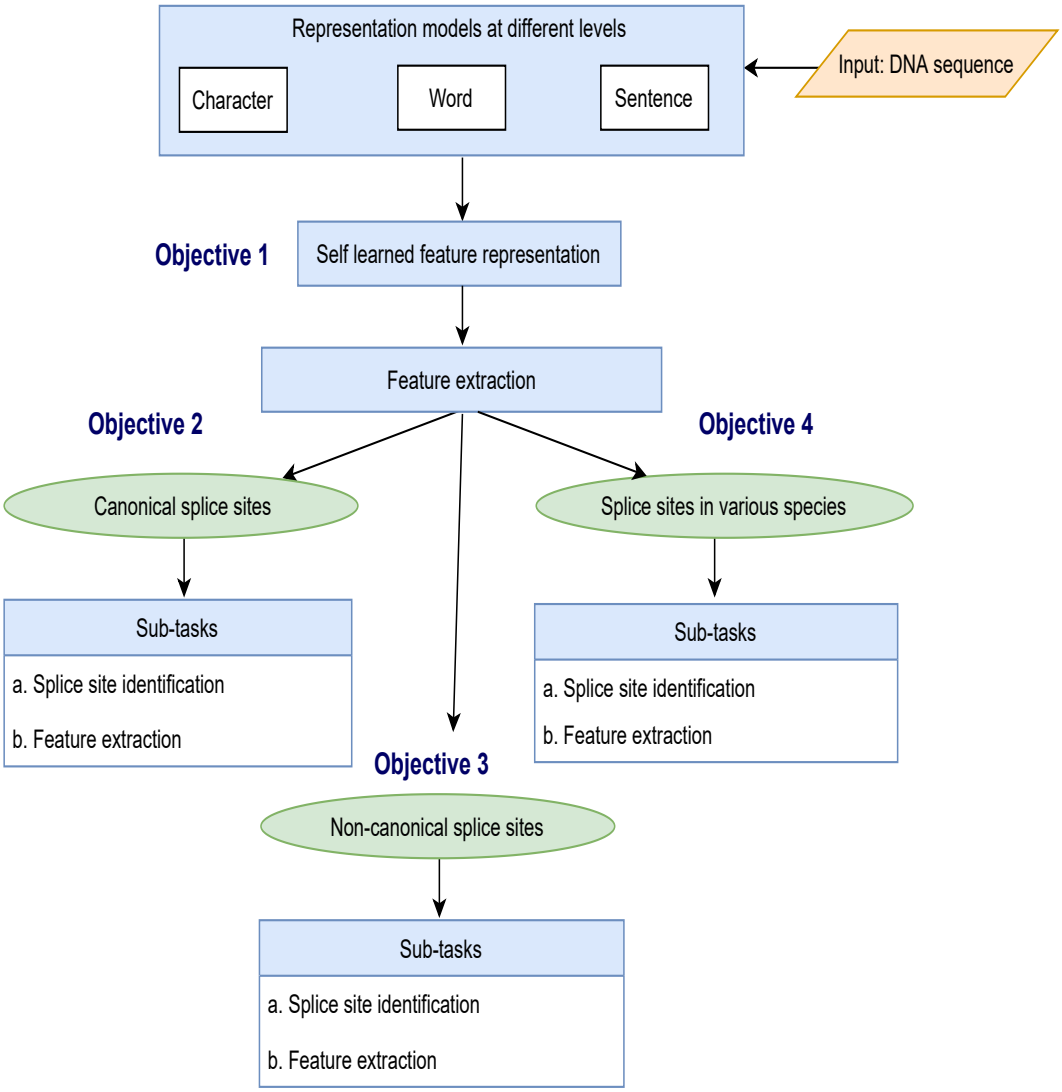


Figure 1.3: Thesis overview

1. SpliceTrans beats state-of-the-art models in identifying canonical and non-canonical splice junctions from human, mouse, and drosophila melanogaster.
2. SpliceTrans also outperforms the state-of-the-art in identifying canonical and non-canonical splice sites from species on which the model is not trained.
3. SpliceTrans maintains its superior performance on imbalanced data making it a more robust choice than its counterparts.
4. We observe that augmenting the training dataset of one species with that of another species improves the performance of the model, especially in identifying non-canonical splice sites. The superior performance of SpliceTrans with such an augmented dataset makes it a favourable choice for annotating poorly studied species using training data from extensively annotated species.
5. We further extract the biological features learnt by the model from non-canonical

splice sites of different species and validate them with the existing literature.

The overview of the thesis contributions is shown in Figure 1.3. Objectives 1 to 4 are chronologically associated with the contributions mentioned above.

1.3 Outline of the Thesis

The thesis comprises eight chapters. The chapter-wise organization of the thesis is given below:

Chapter 1: This chapter briefly touches upon the motivation behind this research, followed by the problem formulation. It also briefly describes the key contributions of this thesis.

Chapter 2: In this chapter, a biological background has been presented for introducing the biological concepts required for a deeper understanding of the problem formulation and contributions of this thesis. We also elaborate on the relationship between cell variables and several disease risks and how developing a deeper understanding of the genomic structure can help unravel the underlying disease risks.

Chapter 3: This chapter is a survey of the prior works based on the application of computational models for identifying splice sites in genome sequences. The chapter presents the evolution of computational models from hand-engineered feature inputs to featureless inputs. It also describes the major challenges faced by the computational models in identifying the splice sites. Furthermore, the limitations of the existing works and the scope of improvements are also identified in this chapter.

Chapter 4: To address the issues of hand-engineered features being neither exhaustive nor optimal, this chapter describes the proposed model (SpliceVec) and how it resolves this issue by learning features from the genome sequence de novo. It also describes the limitation of SpliceVec, which is the motivation behind the research presented in the next chapter. The content used in this chapter is present in the reference [38] co-authored with Tushar Dubey, Kusum Kumari Singh, and Ashish Anand.

Chapter 5: This chapter elaborates the application of SpliceVisuL in the identification of canonical and non-canonical splice sites. The method of extracting the features learnt by SpliceVisuL by applying several visualization techniques is also described in this chapter. The content of this chapter is based on reference [37] co-authored with Aman Dalmia, Athul R., Kusum Kumari Singh, and Ashish Anand.

Chapter 6: This chapter mainly focuses on identifying non-canonical splice sites using SpliceViNCI. The chapter also elaborates on the extraction of relevant features governing non-canonical splicing. The optimal length of genome sequences required for the best performance in non-canonical splice site identification is also presented in this chapter. The work reported in this chapter uses the material from the reference [39] co-authored with Kusum Kumari Singh and Ashish Anand.

Chapter 7: This chapter explores the performance of SpliceTrans for identifying splice sites from multiple species. This contribution tests the generalizability and robustness of the applied model. This chapter also discusses the non-canonical splicing features extracted from human, mouse, and drosophila melanogaster.

Chapter 8: This chapter highlights the conclusions derived from this research and summarizes the contributions made. It also touches upon the future scope of this research work.

"Research is to see what everybody else has seen, and to think what nobody else has thought."

Albert Szent-Gyorgyi (1893 - 1986)
Hungarian physiologist

2

Biological Background

In this chapter, we discuss the biological concepts that are required for a better understanding of the contributions of this thesis explained in the subsequent chapters. However, before discussing the biological concepts, we further elaborate on the relationship between cell variables and several disease risks and how developing a deeper understanding of the genomic structure can help unravel the underlying disease risks.

2.1 Why Should we study the genome?

With the increasing global population, deaths caused by genetic disorders are at an all-time high. The frequency of genetic disorders may vary among different populations depending on factors like lifestyle and environment [104]. If we collect evidence of a wide range of common diseases, we discover the underlying heterogeneity in their causation. Tolstoy, in Anna Karenina, penned down very elegantly the uniqueness of human tragedy, saying that "Every unhappy family is unhappy in its own way". We can say that this uniqueness is also reflected in the case of genetic disorders.

Let us consider the example of *Spinal Muscular Atrophy (SMA)*, the leading genetic cause of infant mortality in North America [26]. SMA is caused by missing or damaged *SMN1* gene in a baby's genome. Humans have two copies of the Survival Motor Neuron (SMN) gene, named SMN1 and SMN2, that are nearly identical (Figure 2.1). The major difference between the two copies is a single nucleotide difference at the beginning of exon 7 (C for SMN1 and T for SMN2). Production of fully functional and stable SMN protein depends on exon 7, which is included in SMN1 but generally excluded in SMN2. Therefore, the SMN2 gene alone cannot produce enough quantity of fully functional SMN protein that is necessary for the survival of motor neurons [26].

However, according to the *Muscular Dystrophy Association (MDA)*, there are other forms of SMA that are caused by genes other than the SMN1 gene. Defects in the *IGHMBP2* gene can cause a rare form of SMA called *Spinal Muscular Atrophy with Respiratory Distress (SMARD)*. Another rare form of SMA, *distal SMA*, can occur with varying symptoms and

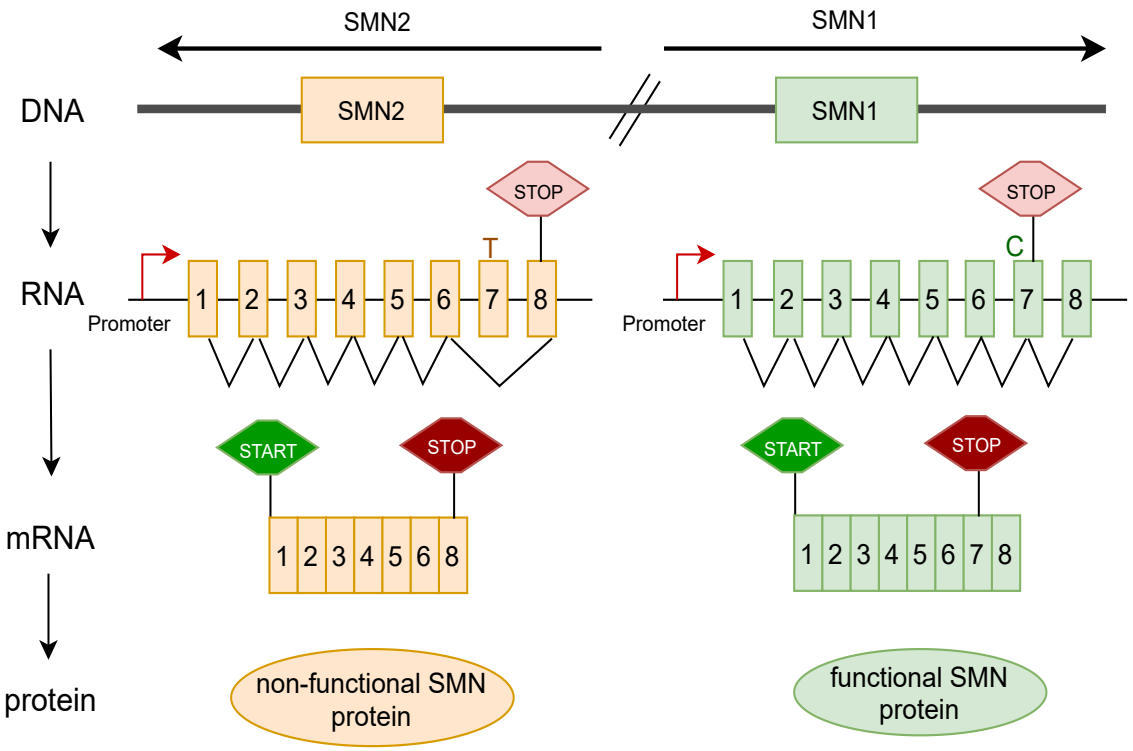


Figure 2.1: A portion of chromosome 5 that contains the two SMN genes

severity depending on the mutated gene. This form of SMA can be caused by several genes, including *UBA1, DYNC1H1, TRPV4, PLEKHG5, GARS, and FBXO38*. SMA severity may also vary depending on factors like *Plastin 3 protein* and *ZPR1 protein* that do not cause the disease but can affect its severity by influencing some related biological pathway. Such factors are called disease modifiers. The quantity of SMN2 genes in the individual can also alter the severity of SMA. Hence, we see that mutations in different genes lying in the same or different pathways can lead to the same disorders. Also, the same mutation may lead to different clinical symptoms (phenotypic traits) in different individuals based on his/her physiological conditions.

Sometimes the same gene can mutate in a hundred different ways causing different diseases in different individuals. Familial genetic studies have shown that naturally occurring mutations in *LMNA* gene are responsible for two groups of apparently unrelated diseases affecting highly specialized tissues: dystrophies of skeletal or cardiac muscles and partial lipodystrophies [125]. In a study [140] it was revealed that two different mutations of the *Shank3* gene produce some distinct molecular and behavioral effects in mice. Later, neuroscientists unraveled that Shank3 is linked to both autism and schizophrenia.

Seeing this causal diversity, we can conclude that causality in this context can rarely be resolved by case-control studies or large scale association studies [86]. To better understand the causes of such heterogeneous diseases, we need to decipher the underlying genomic structure and functionality of an individual. However, the direct association of the genome

(genotype) and the disease traits (phenotype) is critically complex, involving intricate and interrelated biophysical processes shaped through evolution. Therefore, we take an alternative route of associating genotype with molecular phenotype, also called cell variables [76].

2.2 Cell variables and their association with diseases

Cell Variable	Brief Description	Relevance to Disease
Genome annotation	Annotating different regions of the DNA such as marking boundaries of exons and introns, and identifying regulatory sequences.	Change in genomic sequence may render a functional region as non-functional or change its intended function and thus affect regulation.
Binding sites for transcription regulation	Binding proteins to specific regions of DNA which controls the occurrence and rate of transcription.	Variation in binding regions of the DNA can alter whether a gene will be transcribed.
Splicing patterns	Splicing removes introns and chooses exons to be retained during pre-mRNA processing.	Changes to the regulatory elements that affect splicing may change the functionality of the gene.
Cleavage site selection and polyadenylation	The 3‘ signaling region of the transcript is cleaved during gene transcription and a poly(A) tail is added to the mRNA before it is ready for translation.	Alteration to sequence elements can alter cleavage sites which can affect the stability and translation efficiency of the transcript.
RNA structure	RNA folds into three dimensional structure due to intra-molecular interactions.	Modifications in the RNA structure can affect processes that it is involved in, like splicing.
Protein structure	Translation results in sequence of amino acids that fold into a protein. The structure of protein regulates its functions and interactions with DNA, RNA, and other proteins.	Alterations in sequences may lead to mis-folding of proteins which can result in diseases.

Table 2.1: Cell variables related to genomic regulatory mechanisms [Adapted from [76]]

Cell variables are cellular activities that have a closer association with the genome structure and function. Moreover, the cell variables regulate intermediate biochemically active components which can be targets of therapies. Hence developing models to map the genome

to the cell variables can be easier. These cell variables can be observed under different conditions to generate a massive amount of data. These data can be used to train models with high accuracy. Several cell variables, like splicing, transcription rate, polyadenylation, DNA methylation, and protein-nucleic acid binding, have been associated with diseases. Table 2.1 is a compilation of a few such associations. Predicting and analyzing the mechanisms behind the cell variables can provide necessary insights into how genetic variants affect disease risks.

This thesis focuses on one such cell variable called splicing. It has been reported that several fatal human diseases are caused by mutations in core elements of the *splicing* mechanism. For example, *autosomal dominant forms of retinitis pigmentosa* are caused by mutation in the splicing factors *PRPF31/U4-61k* [131] and *PRP8* [20]. Several neurological diseases caused by alternative splicing are stated in [80].

It has been estimated that more than 88% of human protein-coding genes are alternatively spliced during pre-mRNA processing [123]. So, variation in splicing phenomenon leading to disorders is a much frequently observed condition. The association of splicing with diseases makes it further crucial to develop a deeper understanding of the phenomenon for sustaining the health of humankind. Before we discuss the splicing mechanism, let us first understand the genome and gene expression.

2.3 Understanding Genome

A genome is like an instruction book that forms the building blocks of an organism. The genome is composed of deoxyribonucleic acid (DNA). In 1953, DNA was identified as the storehouse of genetic information [129], and by 2001 the Human Genome Project had drafted a raw composition of the entire human genome [70, 71]. However, the challenge of interpreting the structure, function, and meaning of the genetic information persists. Biologist Eric Lander summarized the situation as "Genome. Bought the book. Hard to read."

Information in the genome is stored in the form of genes. Each gene is like an instruction book for making molecules called proteins. However, all genes do not code for proteins. There are about 20,000 protein-coding [32] and 25,000 non-coding [47] genes in the human genome. The protein-coding genes build proteins from amino acid chains through a process called gene expression. Since this thesis focuses on studying the splicing mechanism involved in gene expression, we further discuss gene expression in the biological background.

2.4 Understanding Gene Expression

Gene expression is the process by which information in the genome is synthesized to form functional proteins that control different life activities. Figure 2.2 shows the essential stages that comprise the process of gene expression. It is the most fundamental unit that relates

phenotypes to genotypes.

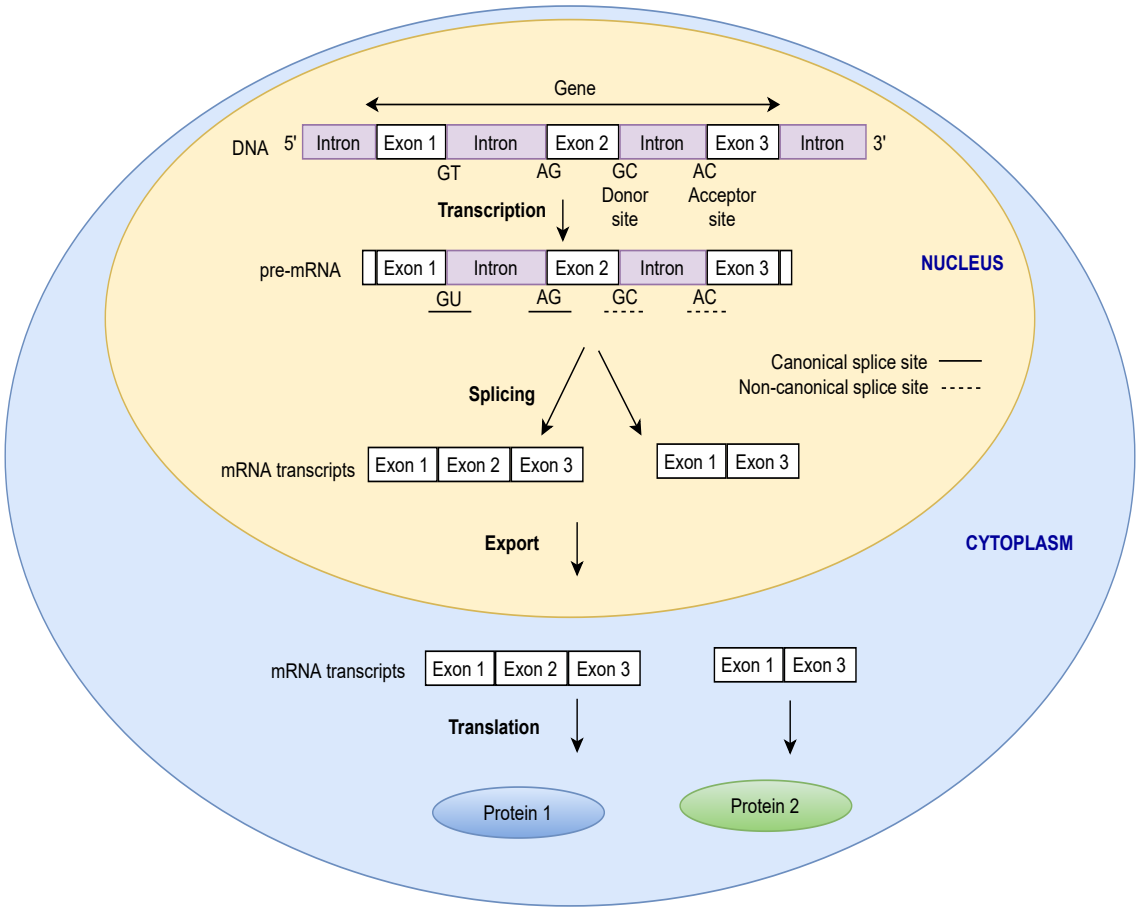
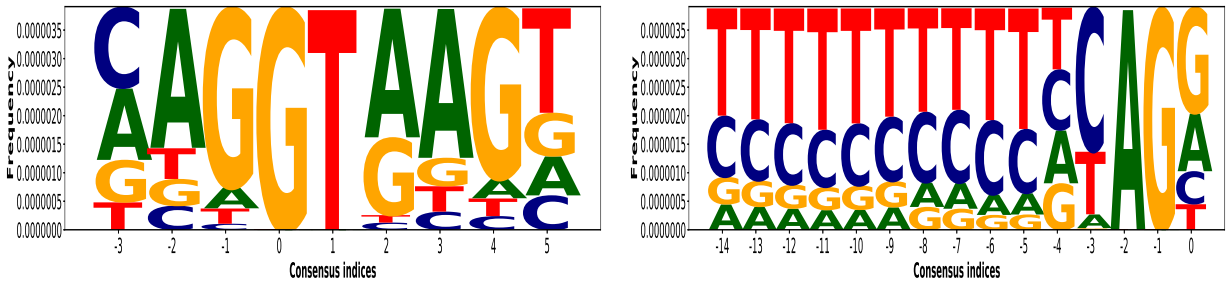


Figure 2.2: An overview of the important stages involved in the process of gene expression.

Gene expression begins with transcription, during which a gene is copied to form a messenger RNA (mRNA). In this stage, the mRNA consists of alternating regions called exons and introns. This version of the mRNA is called precursor mRNA (pre-mRNA), which goes through RNA processing to form a mature mRNA [5].

There are several processes involved in RNA processing that modifies the pre-mRNA before it is ready to be translated into proteins. The pre-mRNA undergoes crucial processes like splicing, polyadenylation, and capping during or after transcription to form a mature mRNA. Finally, the mature mRNA is transported from the nucleus to the cytoplasm, where it is translated to form proteins.



(a) (b)

Figure 2.3: Existing consensus in the canonical (a) donor and (b) acceptor splice junctions of the dataset. The consensus comprises the extended donor site consensus 9-mer $[AC]AGGTRAGT$ and the extended acceptor site consensus 15-mer $Y_{10}NCAGG$. The zero, negative, and positive indices represent the splice junction and its upstream and downstream regions.

2.5 Splicing regulation

Genes in eukaryotes comprise of alternating regions called exons and introns. During or after the transcription, the pre-mRNA transcript goes through the process of splicing. Splicing removes all the introns and ligates all the exons with the help of a complex molecular machine called the spliceosome. Splicing occurs at exon-intron (donor site), and intron-exon (acceptor site) boundaries called splice sites or splice junctions [116].

The splice sites are usually identified by the consensus dimer GT and AG at donor and acceptor sites, respectively. Such splice sites are called the canonical splice sites (Figure 2.2). However, it is also seen that sometimes the splice junctions lack the consensus dimers. Such splice sites are called the non-canonical splice sites. The most commonly observed non-canonical splice sites are characterized by the dimer pairs $GC-AG$ and $AT-AC$ [93].

Apart from the $GT-AG$ consensus, there are other extended donor site and acceptor site consensuses. Figure 2.3 shows the extended donor site consensus 9-mer $[AC]AGGTRAGT$ and the extended acceptor site consensus 15-mer $Y_{10}NCAGG$, known from the existing studies [136]. The acceptor site consensus mostly consists of the polypyrimidine tract (PY-tract) Y_{10}.

2.6 Chapter Summary

This chapter discusses the motivation of this research. The chapter elaborates on the association of genotype with the phenotype. We try to understand the cause of specific phenotypes by understanding the underlying genotype regulating the process. Instead of directly associating the genotype and the phenotype, we use an intermediate layer of molecular phenotypes or cell variables. The cell variables being more closely related to the genotype are

straightforward to study through computational models. Furthermore, the cell variables being associated with active biophysical processes are better targets for therapies.

This chapter further discusses the relationship between different cell variables and several diseases. This establishes the need to study the cell variables in greater detail. We focus on a cell variable called splicing. This chapter throws light on certain biological concepts like genome, gene expression, and splicing mechanism to understand better the contributions described in the subsequent chapters.

"Nature holds the key to our aesthetic, intellectual, cognitive and even spiritual satisfaction."

Edward Osborne Wilson (1929)
American biologist

3

Literature Survey

The exponential increase in genome sequence data demands functional and structural analysis of the data. Annotation of the sequenced data is an essential step for understanding the gene structure [43]. Splice sites are a crucial part of a gene structure since they depict the junctions between the alternating regions of a gene called exons and introns. Therefore, accurate identification of splice sites in the genome sequence is necessary for annotating gene structure and it requires accurate modeling of regions around splice sites.

In this chapter, we discuss the challenges in the identification of splice sites from genome sequences. We shall also discuss the existing research works on the task of identifying splice sites and understanding the splicing mechanism. Subsequently, we shall selectively throw light on some of the limitations of the existing works.

3.1 Challenges in identification of splice sites

The major challenges faced by the computational models in the identification of splice sites are discussed below.

1. **Frequently occurring consensus sequences that are not splice sites:** Since the dinucleotides GT and AG frequently appear in the genome sequence, thus in the actual scenario, the number of negative samples is much more than that of positive samples in a genome. Therefore, to predict true splice sites, the model has to handle the imbalanced positive and negative classes of the dataset [75].

2. **Finding the optimal feature set:** Important features regulating splicing, other than the core splice site consensus dimers, are contained in exonic and intronic regions. These features function by recruiting sequence-specific RNA-binding protein factors that either activate or repress the use of adjacent splice sites. However, it is not guaranteed that all the features important for the splicing activity are already discovered. Moreover, the regulating factors may be located at a remote distance from the splice

sites in the genome sequence [33, 114]. Therefore, it is challenging to decide the length of the sequence required to identify all the splice sites.

3. **False positive and false negative splice sites:** The dinucleotides GT and AG frequently appear in eukaryotic genome sequences, but all of them are not splice sites. This can produce false positives. Also, non-canonical splice sites can be detected as false negatives by the model. Therefore, it can be difficult for the model to identify splice sites by simply looking at the donor and acceptor junctions. To improve the overall performance of gene prediction, it is essential to reduce false predictions of splice sites [43].

3.2 Models for splice site identification

To achieve a deeper understanding of the splicing mechanism, different experimental, statistical, and machine learning based techniques have been employed to date. It is realized that computational tools identifying the splice sites and the splicing features offer an exciting alternative to experimental methods since they are faster and reduce the search space for experimental verification. However, the computational tools have limitations due to the arbitrary nature of threshold scores, challenges in prioritizing one tool over the other, and lack of reliable standard interpretation guidelines [114]. Therefore, both experimental and computational models need to work hand in hand to achieve the target. Next, we shall discuss the two important types of models used in this field.

3.2.1 Sequence alignment-based and statistical techniques

Several sequence alignment-based and statistical models have been used to study splicing of genes and its corresponding implications on gene expressions and disease traits. The alignment-based strategies [44, 124, 126] rely on reconstructing exons by mapping millions of short RNA reads to the reference genome sequence. Splicing is then identified when two sections of a read map to two different exons of the reference genome, separated by the intronic region in between. However, in the alignment-based methods, there is a possibility of a short read randomly matching a large reference genome containing multiple occurrences of the short read sequence [78]. Moreover, the existing alignment-based methods [11, 124] consider only canonical splice junctions in the prediction task [75].

Katz et al. [62] developed a statistical model, named the mixture-of-isoforms (MISO) model, that estimates expression of alternatively spliced exons to infer isoform regulation from high-throughput sequencing of cDNA fragments (RNA-seq). Estimations of the alternative splicing levels improved significantly on incorporating mRNA fragment length distribution in paired-end RNA-seq. Barash et al. [15] formulated a splicing code as a statistical inference problem to draw a relationship between genomic features specific to cellular conditions and the exon splicing patterns. The amount of splicing information accounted for

by the code was measured by an information theory based formula of code quality. To aid the assessment of functional consequences of variants near splice sites, expression minigenes (EMGs) were created by Sharma et al. [114]. These minigenes helped determine the RNA and protein products generated by splice site variants implicated in cystic fibrosis. This paper refers various tools used to assess the effect of variants upon the strength of splicing and to predict the splicing isoforms using EMGs. The tools used were both statistical and machine learning based.

3.2.2 Machine learning based techniques

Machine learning techniques have been extensively applied in this field because of their efficiency. We shall discuss the significant successes of machine learning models in this domain.

There are several instances of the application of neural networks in splice site prediction. Hatzigeorgiou et al. [48] used separate neural network modules for distinct signals like coding regions, splice sites, and transcription start regions. They used back-percolation, cascade correlation, and time-delay neural networks, which performed better than the traditional backpropagation algorithm. Reese et al. [106] presented an improved version of the generalized Hidden Markov Model (GHMM) based genefinder named Genie. They replaced the existing splice site sensors in Genie with two novel neural networks based on dinucleotide frequencies which showed significant improvements in the sensitivity and specificity of gene structure identification. A complementary encoding method where the nucleotide A was represented by 1, T by -1, C by 2, and G by -2 was proposed in [25]. Compared with the traditional encoding method, this complementary encoding method significantly reduced the training time of the network and produced fewer incorrect non-splice sites. Neural network-based techniques for splice site prediction were surveyed in many review papers like [42] and [98]. [120] and [113] have used support vector machine (SVM) with appropriately designed kernels like locality improved kernel, polynomial kernel, and weighted degree kernel to distinguish between true and false splice sites. SVM proves to be a great choice in classification problems, and it produced promising results in this field compared to hidden Markov models [50, 133, 135].

Combining different techniques has proved to produce better results in this field. This can be seen in many research efforts to date. The first and second order Markov model was combined with backpropagation neural network in [51]. Zhang et al. combined Bayes kernel with SVM in [137]. Neuro-fuzzy network and clustering were combined in [91] for splice site prediction. Wei et al. [130] presented a splice site prediction method based on the first order Markov model for pre-processing the DNA sequence and SVM for classifying the splice sites. A similar combination of techniques was also used much before in [17]. These models showed promising results, but the first order Markov model could not model the codon composition of exons. This limitation was handled in [43] by using the second order Markov model for pre-processing the sequences. This model showed improvement over most of the existing splice site predictors.

Deep neural networks have also been used for splice site prediction. Lee et al. [74] employed a deep recurrent neural network for identifying canonical splice sites. [139] and [142] applied deep convolutional neural networks for identifying splice junction pairs and splice junctions, respectively. Lee et al. [75] proposed a deep belief network-based methodology for identifying canonical and non-canonical splice sites. Jaganathan et al. [59] applied a deep residual neural network to identify if a genomic position is a donor, acceptor, or neither.

The methods discussed above can be summarized as given in the following table (Table 3.1).

Approach	Novelty	Features	References	Year
Hidden Markov Model	Capturing higher order dependencies in genome sequences	Consensus and degeneracy features	[50, 133, 135]	2001-2009
Neural Network	Preprocessing of input sequences to capture sequence features via a sliding window of nucleotides	Sequence features	[25, 42, 48, 52, 81, 96, 98, 106, 109]	1996-2013
Support Vector Machine	Appropriate and novel kernels for SVM	Positional (most common), compositional and dependency features	[87, 113, 120]	2007-2016
Combined models	Choice of better preprocessing method combined with appropriate classification tool	Depends on the classifier being used	[17, 43, 91, 130]	2006-2015
Deep Networks	Deeper networks where hidden variables represent sequence features	Sequence features	[59, 74, 75, 77, 139, 142]	2014-2019

Table 3.1: Machine learning approaches for predicting the splicing regulations

3.3 Features for splice sites identification

3.3.1 Hand engineered feature set

The prediction of splice sites is facilitated by identifying relationships and dependencies among the nucleotides around the splice sites. This is motivated by the observation that the splicing signals are most likely to reside in the vicinity of splice sites [3]. The learning

algorithms need a set of features for training the model. Designing feature sets that best represent the dataset has always remained a challenge for splice site prediction. The presence or absence of specific nucleotide sequences close to the splice sites was considered as features for splice site prediction for a long time [17, 22, 34, 56, 101, 106, 120].

Pertea et al. applied Markov models to capture dependencies among the neighbouring nucleotides in the region around splice sites [101]. Reese et al. counted dinucleotide frequencies to design the feature set [106]. Degroeve et al. used positional information like count of mono, di, and trimers near the splice sites as input to an SVM classifier [34]. They also used compositional context using trimers to hexamers. Other works like [137] and [28] have also used positional features for linear SVM classifier and Bayesian Networks. Since all the splicing signals are still not known, the hand-engineered features may adversely affect the accuracy of prediction models due to the inclusion of irrelevant features as well as high dimensionality. There have been attempts to refine and optimize the features as well as include more relevant features by taking into account recent experimental observations.

3.3.2 Optimized feature set

The splicing phenomenon is driven by a range of splicing signals distributed in the genomic region near the splice sites. Researchers have worked with more than a thousand features to predict the splicing pattern [15, 132]. The limited knowledge of the splicing signals leads to the inclusion of irrelevant features, which can affect the performance of predictive models. Moreover, they add to the complexity of the problem by increasing the dimensionality of the feature set. The identification of relevant biological features becomes very important in such large and complex problems. Therefore, feature selection comes to the rescue. Feature selection helps attain good or even better solutions using a restricted subset of features and a faster classification. Thus, extracting knowledge from complex biological data using robust and fast feature selection methods are of key importance [110].

A wrapper-based feature subset selection algorithm was used in [33] along with support vector machine for selecting features relevant for splice site prediction. Results showed that the selected features performed better than all features taken together. Saeys et al. presented a novel method for feature subset selection based on the estimation of distribution algorithms, a more general genetic algorithm framework. They applied it for splice site prediction [110]. They first ranked the features then iteratively discarded the less important features. They showed that it could be used to gain insight into the underlying biological process of splicing. A similar objective was fulfilled in several other research works where feature subset was selected from a wide range of features to improve predictive performance [58, 111, 112]. From this survey, we can derive that it is important to cover as many features as possible for better results, but at the same time, the set of features should be optimal. The use of irrelevant or less important features will only add to the dimensionality and complexity of the problem without improving the results.

3.3.3 Self-learnt feature set

The limited biological knowledge and increasing need for feature selection motivated the adoption of a new approach. It was now desired that a learning model should extract relevant splicing features without any manual feature engineering. Instead, the model will capture motifs that act as splicing signals for splice site selection from the biological sequences by itself.

Lee et al. [75] proposed a deep Boltzmann machine based methodology for splice junction prediction. Zhang et al. have employed a deep convolutional neural network (CNN), named *DeepSplice* [139], that learns features that characterize the true and decoy splice junctions. They have predicted novel splice junctions based on these features and obtained state-of-the-art performance of 96% accuracy. Zuallaert et al. [142] also applied a deep convolutional neural network (CNN) to identify canonical splice sites.

3.4 Limitations of the current literature

The limitations of the current state-of-the-art models in identifying splice sites are discussed below. The limitations mentioned here are addressed in the contributions of this thesis.

3.4.1 Extraction of splicing features

Several deep and shallow neural network models have been successfully applied on various biological sequence-based tasks like gene expression regulation [6, 8, 73], protein classification [9, 53], and protein structure prediction [10, 40, 55]. The recent trend in genome sequence analysis is applying neural network models that learn features from the sequence de novo [79, 89]. The primary motivation to let the model learn relevant features by itself is to avoid the existing knowledge bias. However, the inference of biologically relevant information learnt by the models remains a challenge across the domain of genome sequence analysis. Several visualization techniques have been effectively applied across various domains to analyze learning models and infer relevant features.

To decipher the reason behind the outstanding performance of various learning models, attempts have been made to monitor the change in model weights as learning progresses [73, 75]. Angermueller et al. [8] extract sequence motifs by aligning sequence fragments that maximally activated the filters of the convolutional layer for predicting single-cell methylation states. Park et al. [100] perform one-dimensional global average pooling on the attention weighted output of an RNN to discover the parts of the sequence that are significant for identifying pre-miRNAs. Lanchantin et al. [69] explore various sequence-specific and class-specific visualization techniques to obtain the important nucleotide positions present in a genome sequence to classify transcription factor binding sites.

Visualization techniques have also been applied to decipher the 'black box' nature of deep learning models identifying splice sites. Zuallaert et al. [142] apply a back-propagation based visualization technique, called *DeepLIFT* [117], over a deep convolutional neural network (CNN) to infer features relevant to splicing. Zhang et al. [138] employ deep CNN for the identification of canonical and semi-canonical [24] splice junctions. They further use deep Taylor decomposition [92] to assess the contribution of nucleotides in the classification decision. Jaganathan et al. [59] annotate the complete RNA transcript by classifying each nucleotide as a donor, acceptor, or neither. They apply dilated convolution layers to enable the learning of sequence determinants from thousands of flanking nucleotides. The authors in [59] further extend the model to evaluate the effects of genetic mutations on splicing. They eventually enhance the understanding of the relationship between mutations in the genome and various human diseases. However, these models predict either the donor or the acceptor splice sites based on the dataset used. Also, they have considered only canonical splice junctions for visualizing the important genomic regions.

3.4.2 Identification of non-canonical splice sites

Most of the existing computational methods either consider only the canonical splice sites for the prediction task or perform relatively poorly in identifying non-canonical splice sites. However, identifying non-canonical splice sites is also equally essential to understand splicing [75]. Understanding non-canonical splicing events can help explain the unconventional regulation pathways for which a function has not yet been identified.[63] Non-canonical splicing most often has a role in regulating gene expression.[118] Some of the non-canonical splice sites are crucial in cellular activities like immunoglobulin gene expression and other critical biological events [23]. Mutations in the genomic regions far away from canonical splice sites can cause disease by disrupting the splicing mechanism and activating non-canonical splice sites via non-canonical splicing. Finally, non-canonical splicing mechanisms can be targeted or exploited for new therapeutic strategies. Such strategies have already been applied to diseases like cancer, muscular dystrophies, and haemophilia.[27, 66]

The emergence of deep learning era led to the application of models which learn the complex splicing signals from the genome sequence *de novo*. There are implementations of deep Boltzmann machine [75], deep convolutional neural networks (CNN) [138, 139, 142], distributed representation learning [38], recurrent neural networks (RNN) [37, 74], and deep residual neural network [59] for the prediction of splice junctions. Some of these models are based on the identification of only canonical splice junctions [59, 74, 138, 142]. The models which include non-canonical splice junctions ([37, 38, 75, 139]) consider limited splicing context at the junctions. These methods extract upstream and downstream sequences at both donor and acceptor junctions, ranging from 30 to 40 nucleotides (nt) since this length is considered optimal for splicing signals in various studies [101, 106]. Also, these models target achieving the overall optimal performance considering both canonical and non-canonical splice junctions together.

3.4.3 Identification of splice sites in multiple species

Most of the current neural network models identify splice sites from only a single species [37, 38, 59, 74, 75, 138, 139]. A few research works like [4, 127, 142] identify splice sites from multiple organisms, namely Homo sapiens, Arabidopsis thaliana, Oryza sativa japonica, Drosophila melanogaster, and C. elegans. However, these researches have some limitations discussed below.

Zuallaert et al. [142] proposed a prediction model named SpliceRover that identifies splice sites from human and arabidopsis thaliana. However, they do not discuss the generalization capability of their proposed model by training on one species and testing on another. Albaradeia et al. [4] proposed a model named Splice2Deep, which trains and tests on different species and still identifies the splice sites with high accuracy. However, they do not extract or discuss any biological features learnt by the model. Wang et al. [127] proposed SpliceFinder, which trains a CNN model to identify acceptor, donor, and false splice sites. Although SpliceFinder trains and tests the model on both canonical and non-canonical splice sites of various species, it extracts and visualizes features from canonical sites only. Furthermore, the performance of SpliceFinder is tested on the identification of splice sites extracted from the same version as that of the training dataset. Hence, the research is not focused on the identification of novel splice sites. However, the ability to identify novel splice sites indicates that a model is generalizable.

3.5 Chapter Summary

This chapter discusses the major challenges of identifying splice sites from genome sequences. The presence of non-canonical splice sites can increase the number of false negatives. Similarly, the frequent presence of the dimer pair $GT - AG$, which is the same as the consensus dimer but not annotated as actual splice sites, can increase the number of false positives. The frequent occurrence of consensus dimer in the genome also increases the ratio of negative to positive samples. This yields an imbalanced dataset in the actual scenario and poses a challenge for the prediction models.

Several sequence alignment-based, statistical, and machine learning based models have been proposed for identifying splice sites from genome sequences. Traditionally, machine learning based models take hand-engineered feature sets as inputs. However, such manually extracted feature sets are neither optimal nor exhaustive. This increases the dimension of the model's input which can, in turn, degrade the model's performance. There is also the possibility of missing out on important regulatory features in the hand-engineered input.

The increasing availability of sequenced data in the current era has motivated the use of machine learning models which can learn features from the genome sequences de novo. This has reduced the dependency on manual extraction of features to a great extent. However, inherently such machine learning models have a 'black-box' characteristic. This means that it is a challenge to extract the features learnt by the model, which plays a crucial role in the

decision-making of the model.

Several visualization techniques have been applied for the extraction of the relevant features learnt by the model. However, most of the existing models extract features related to canonical splicing only, although it is equally important to study features regulating non-canonical splicing for a comprehensive understanding of the splicing phenomenon. Furthermore, the models primarily identify splice sites from a single species only. However, the capability of identifying splice sites from multiple species can establish the robustness and generalizability of the model.

"Nature holds the key to our aesthetic, intellectual, cognitive and even spiritual satisfaction."

Edward Osborne Wilson (1929)
American biologist

4

SpliceVec: distributed feature representations for splice junction prediction

4.1 Introduction

This contribution is targeted towards the first objective of the thesis. We aim to identify splice sites using neural network models without any hand-crafted features as input. Instead, the model learns features by itself from the genome sequences. This work [38] introduces a novel approach for distributed feature representation of splice junctions by embedding it in an n-dimensional feature space. Each dimension in the feature space represents one feature of the corresponding splice junction. This embedding, named *SpliceVec*, is in the form of n-dimensional continuous distributed vector representation. The embeddings are learnt by a shallow neural network using unsupervised data.

We explore two variants of SpliceVec, namely *genome based SpliceVec (SpliceVec-g)* and *splicing-context based SpliceVec (SpliceVec-sp)* for feature representation of splice junctions. We evaluate the quality of SpliceVec in both intrinsic and extrinsic tasks. For the intrinsic evaluation, we visually inspect two dimensional representation of true and false splice sites using Stochastic Neighbor Embedding (t-SNE) [84]. We evaluate SpliceVec on splice junction classification task for the extrinsic evaluation. In contrast to the recent deep learning methods, we use simple multilayer perceptron (MLP) as a classifier. We name this model as *SpliceVec-MLP*.

Our results and contributions can be summarized as follows:

- We propose two variations (SpliceVec- g and SpliceVec- sp) for feature representation of splice sites. SpliceVec outperforms state-of-the-art methods by 2.42-18.86% in terms of accuracy for splice site prediction.
- We explore the optimal sequence length that best captures the splicing signals for improving the prediction results. We find that inclusion of entire intronic sequence significantly boosts the predictive power of the classifier.

- The proposed feature representations are more robust in handling reduced training samples. SpliceVec maintains an accuracy above 99% even with a 60% reduction of training samples whereas the accuracy of its counterpart drops by about 6%.
- SpliceVec is more consistent in its performance with class-imbalanced data making it more suitable for the real scenario where number of pseudo sites are several times more than that of true splice sites.
- SpliceVec-MLP identifies non-canonical splice junctions with 100% accuracy indicating that our feature representations are invariant to both canonical and non-canonical splice junctions.
- SpliceVec-MLP can be deployed in both CPU and GPU environment. SpliceVec-MLP, being 12.94 times computationally faster than the state-of-the-art model, contributes as a suitable option for classification of the abundant annotated sequences available these days by high-throughput sequencing technologies.

4.2 Methods

The proposed approach can be divided into two stages: the feature representation stage that generates a distributed representation for each splice junction based on either of the two frameworks, namely *word2vec* and *doc2vec*, and classification of splice junctions using MLP. We shall discuss all these in the following subsections. An overview of the proposed approach is shown in Figure 4.1.

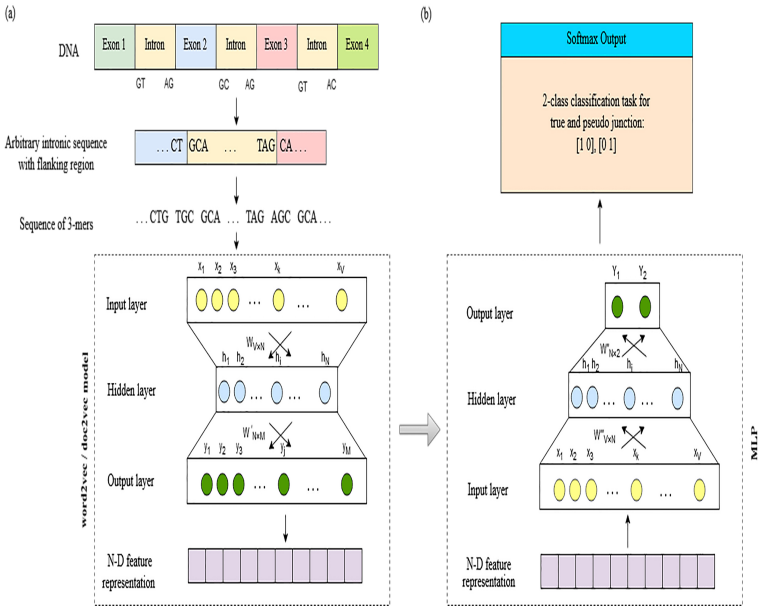


Figure 4.1: Proposed approach: (a) feature representation; (b) splice junction classification.

4.2.1 Data

We have used the GENCODE annotation [47] (version 26), based on human genome assembly version *GRCh38*, for extracting true and false splice junctions. This version was released in March 2017. We extract 294,576 splice junctions from the protein coding genes. Based on these splice junctions, we observe that an intron length varies from 1 to as much as 1,240,200 nucleotides (nt). A recent study suggests that the shortest known eukaryotic intron length is 30 base pairs (bp) belonging to the human *MST1L* gene [102]. Introns shorter than 30bp are usually accounted to sequencing errors in genomes. Based on this study, we consider only those introns whose length are greater than 30 bp. This reduces the number of splice junctions to 293,889. We select 293,889 false splice junctions by randomly searching for splice site consensus sequences GT and AG which are not annotated as splice junctions. This is considered as a necessary condition for selection of false splice junctions because more than 98% of splice junctions are canonical, that is, they contain the consensus dinucleotide GT at the donor site and AG at the acceptor site [24]. In DeepSplice, the length of false splice junctions is considered to be lying between 10 and 300,000 nt, the reason of which is not clear. So, instead, we consider only those false splice junctions whose length is not less than 30 nt and not more than 1,240,200 nt with both donor and acceptor splice sites lying in the same chromosome. We consider this range to mimic the scenario of true introns.

4.2.2 Distributed representation

Vector representation of a word or a sentence is an integral part of many natural language processing (NLP) tasks. Although local representations, like N-grams and bag-of-words, have been successfully applied in NLP, they lack efficiency due to sparsity and high dimensionality. Most importantly, such representation needs to be defined explicitly. In the recent times, distributed representation has been most successful in complex NLP tasks. This type of representation is learnt based on the connection and interaction between words appearing in various contexts within a chosen corpus.

With this idea, Mikolov et. al. proposed word2vec models [88] that compute continuous vector representations for words learnt by shallow neural networks. These models embed each word in an n-dimensional space where syntactically and semantically similar words appear close to each other. The model has two different architectures- continuous bag of words (CBOW) and skip-gram. CBOW predicts the current word given some surrounding context words whereas skip-grams predicts context words given the current word.

This model has been further extended in the form of doc2vec model [72] to incorporate continuous vector representations for variable length texts like sentences, paragraphs and documents. This model also comes in two architectures- distributed bag of words (DBOW) and distributed memory (DM). DBOW, similar to skip-gram architecture of word2vec model, predicts the context words from the document vector. DM, on the other hand, predicts the current word based on surrounding context words as well as the document vector. This is similar to CBOW architecture of word2vec model. In doc2vec model, the word vectors are

global and shared among documents whereas the document vectors are local to the document and learnt only in context of the corresponding document.

4.2.3 Distributed representation of splice junctions

Both the models, described in the previous section, demand as input a large corpus of text and produce outputs in n-dimensional vector space where each unique word in the corpus is assigned a vector in that space. A corpus in NLP is a continuous chain of words following certain grammatical structure. On the other hand, biological sequences are a continuous string of four characters, A, C, G, and T, representing nucleotide bases Adenine, Cytosine, Guanine and Thymine respectively. Our corpus also contains N which can represent any of the four nucleotides. Since there is no concept of words in case of biological sequences, it can be broken down into k-mers of any length k. There have been experiments with variable length k-mers [97] as well as both overlapping and non-overlapping k-mers [9, 64] for different bioinformatics problems. In this work, we focus on overlapping 3-mers. For example, a biological sequence ATTGGCA yields the following sequence of words: ATT, TTG, TGG, GGC, and GCA. Thus, our vocabulary can have a maximum of $5^3 = 125$ distinct words. After this pre-processing step of fragmenting biological sequences into words, we feed the corpus into both word2vec and doc2vec models to generate n-dimensional embedding, named SpliceVec. SpliceVec-g is based on word2vec model whereas SpliceVec-sp is based on doc2vec model.

4.2.3.1 Genome based SpliceVec

This type of feature representation is generated using word2vec model. For word2vec model, our corpus is the complete human genome assembly (version *GRCh38.p10*). We break down the genome assembly into chromosomes. We consider 24 chromosomes, that is, chr1 to chr22 as well as X and Y chromosomes. We have excluded the mitochondrial and unlocalized sequences because of its exceptionally small sizes. We further fragment each chromosome into sentences of 2000 characters. Each sentence is broken into overlapping 3-mers representing words. We train word2vec model with the complete sequence of 3-mers. The word2vec model produces an n-dimensional vector representation for each unique 3-mer in the corpus. We next compute the vector representation of a splice junction by taking the average of summation of the vector representation of each word in the splice junction sequence. We used gensim [107] library of python to generate the word vectors.

4.2.3.2 Splicing-context based SpliceVec

This variation of feature representation is based on doc2vec model. For doc2vec model, our corpus is the complete set of true and false splice junctions. Each splice junction, considered as a document, is broken down into overlapping 3-mers to form words and fed into doc2vec model as training data. The model generates a vector representation for each

unique word and each splice junction sequence in an n-dimensional hyperspace. We use the C-implementation of doc2vec model by Le and Mikolov [72] for generating the vectors.

Both the CBOW and skip-gram architecture for word2vec model, as well as the DBOW and DM architecture for doc2vec model are illustrated in Figure 4.2 considering an arbitrary biological sequence ATTGGCA. The sequence is broken down into 3-mers sequence ATT, TTG, TGG, GGC, and GCA where TGG is the current word and remaining four words are context. ID refers to the splice junction which contains the given sequence of 3-mers.

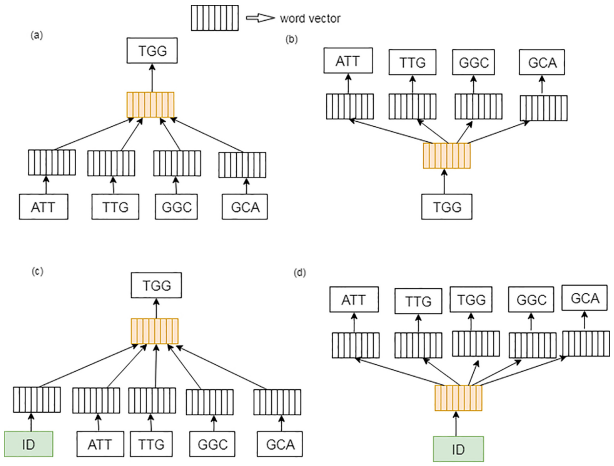


Figure 4.2: Architecture of word2vec and doc2vec models. (a) CBOW for word2vec (b) skip-gram for word2vec (c) DM for doc2vec and (d) DBOW for doc2vec.

4.2.4 SpliceVec feature space construction

We generate both SpliceVec-g and SpliceVec-sp for each of 587,778 splice junctions, consisting of 50% true and 50% decoy splice junctions. There are several hyper-parameters for both the word2vec and doc2vec models based on which SpliceVec is generated and each of these hyper-parameters needs to be tuned as per the underlying task to be accomplished. We generate several sets of SpliceVec by variations and different combinations of the hyper-parameters mentioned in Table 4.1. We evaluate the embeddings on the classification task of splice junctions. We partition 30% of our dataset as test data. Out of the remaining 70%, 20% is used as validation set for tuning the hyper-parameters of MLP for classification. We train the MLP with the training data in a Tensorflow [1] implementation. Based on the performance of our classifier, we fix the hyper-parameters as given in Table 4.1. Default values of respective models are considered for other hyper-parameters not mentioned in the table.

Hyper-parameters	word2vec	doc2vec
Training method	CBOW	DM
Vector Size	100	100
Window Length	5	5
Min Count	5	1
Negative Sampling	5	5
Iterations	15	15

Table 4.1: Optimal hyper-parameters for word2vec and doc2vec training models.

4.2.5 Classification of SpliceVec by MLP

We use an MLP with one hidden layer as the classifier for splice junction classification. We take each n-dimensional SpliceVec representing one splice junction as the input data and send the weighted input to each node of the hidden layer. Each node j of the hidden layer receives the signal (x_i) from each node i in input layer multiplied with a weight (w_{ji}). The effective signal S_j of a node j is:

$$S_j = \sum_{i=0}^{n} w_{ji} x_i$$

where n is the number of nodes in input layer and x_0 is the bias. The signal, in hidden layer, undergoes the activation function. We use rectified linear units (ReLU) as the activation function. ReLU sets any negative input signal (x) to zero by the following function:

$$f(x) = max(0, x)$$

In the output layer, weighted sum of outputs from hidden layer undergoes the softmax activation function that gives the class probabilities of the input. There are two nodes in the output layer corresponding to true and false splice junctions. The predicted output is compared to the expected output using cross entropy as the loss function. The cross entropy loss can be defined as follows:

$$H_{y'}(y) = -\sum_{i} y'_i log(y_i)$$

where y is the predicted output and y' is the expected output. We use Adam Optimizer [65] to minimize the loss by updating weights at each layer.

We compare classification performance of SpliceVec-MLP with existing state-of-the-art approaches for splice site prediction, named SpliceMachine [34] and DeepSplice. SpliceMachine is based on linear support vector machines (LSVM) whereas DeepSplice is based on CNN. We tune hyper-parameters for DeepSplice, SpliceMachine and SpliceVec-MLP by partitioning training data into train, test and validation set as explained in the previous subsection. The optimal hyper-parameters are given in Table 4.2. All the experiments are carried out on a 3.20 GHz Quad-Core Intel Core i5 with 8GB memory. We evaluate the performance

of the classifier based on precision, recall, accuracy, and F1 score.

DeepSplice	
Batch size	160
Dropout keep_prob	0.8
Learning rate	0.001

SpliceMachine	
Context size (donor)	20, 60
Context size (acceptor)	20, 20
C parameter for LSVM	0.015625

SpliceVec-MLP	
Batch size	128
Hidden nodes	1024
Learning rate	0.001

Table 4.2: Optimal hyper-parameters of the classifiers.

4.3 Results

4.3.1 Qualitative analysis of SpliceVec

To analyze the quality of embeddings, we plot both SpliceVec-g and SpliceVec-sp by projecting it from a 100-dimensional feature space to a 2D space using t-SNE. As a comparison, we plot an equal number of randomly generated 100-dimensional vectors that are randomly assigned as true or false splice junctions. The 2D embeddings clearly show that both SpliceVec-g and SpliceVec-sp form clusters which suggest that they capture the features specific to true and false splice junctions (Figure 4.3).

We see that SpliceVec-sp displays better partitioning of true and false junctions compared to SpliceVec-g. This is because SpliceVec-sp, based on doc2vec model, captures the order in which words appear in the sequence, thus generating more meaningful feature vector. In Figure 4.3, points in red represent false splice junctions whereas points in blue represent true splice junctions.

4.3.2 Optimal sequence length for SpliceVec

We vary the length of flanking region at the donor and acceptor splice sites to find the optimal length of sequence that produces the best results in classification. These flanking regions are considered in order to capture the splicing signals present in vicinity of the intron boundary. For each splice junction, we first extract only 10 nt from upstream and downstream sequence of both donor and acceptor splice sites, thus yielding a sequence length of 40 nt. We increase the length of flanking regions upto 40 nt, in an interval of 10 nt. We also extract entire intron sequence along with 10 nt upstream and 10 nt downstream of donor and acceptor splice sites, respectively. We have taken the entire intron sequence as part of the input because there are evidences of intronic sequences, more than 150 bp long, being

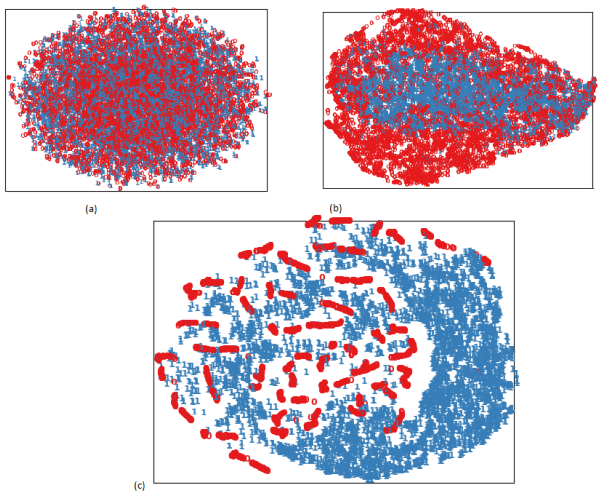


Figure 4.3: t-SNE plots for different embeddings. (a) Random embedding (b) SpliceVec-g (c) SpliceVec-sp. Each point represents a splice junction. Points in red represent false splice junctions whereas points in blue represent true splice junctions.

conserved around the alternative exons [121, 128], thus making intronic elements more important in regulating AS. In this case also, we increase the length of flanking region from 10 nt upto 40 nt in an interval of 10 nt.

Table 4.3 demonstrates the results obtained on varying the sequence length. We indeed observe that the improvement in performance of classifier is significantly more when we consider the entire intronic sequence with 10 nt flanking region compared to increasing the length of flanking upstream and downstream regions at both the donor and acceptor splice sites. On further increasing the length of flanking region, in the case where full intron was considered, we observe that performance degrades for SpliceVec-g. This indicates that irrelevant features are being captured as the length of flanking region is increased. On the other hand, SpliceVec-sp is consistent even with increased flanking region. This is because SpliceVec-sp, based on doc2vec model, provides more robust embeddings especially with longer documents and are therefore less sensitive to irrelevant features. We therefore perform our classification task on SpliceVec generated from full intron with 10 nt flanking regions for further analysis. The problem of having very long and variable length introns as input is solved by the fact that each splice junction will be reduced to a 100 dimensional SpliceVec, which is much less than the actual length of intron.

Sequence length	SpliceVec-g	SpliceVec-sp
	$Ac, Pr, Re, F1(\%)$	$Ac, Pr, Re, F1(\%)$
10nt flanking	82.26, 80.61, 85.02, 82.73	99.77, 99.80, 99.73, 99.77
20nt flanking	81.09, 78.85, 85.01, 81.79	99.55, 99.50, 99.61, 99.55
30nt flanking	82.68, 80.67, 85.98, 83.22	99.52, 99.45, 99.58, 99.52
40nt flanking	84.81, 82.87, 87.80, 85.24	99.37, 99.33, 99.41, 99.37
10nt flanking + intron	**98.15, 98.40, 97.89, 98.14**	99.88, 99.81, 99.95, 99.88
20nt flanking + intron	97.73, 97.97, 97.48, 97.72	99.94, 99.92, 99.96, 99.94
30nt flanking + intron	97.35, 97.86, 96.82, 97.34	99.93, 99.92, 99.93, 99.93
40nt flanking + intron	97.04, 97.30, 96.77, 97.03	**99.97, 99.97, 99.97, 99.97**

Table 4.3: Performance of splice junction classification using SpliceVec-g and SpliceVec-sp by varying length of junction sequences. We compute accuracy (Ac), precision (Pr), recall (Re) and F1 score (F1) in percentage as performance measures. The values are average of five simulations.

4.3.3 Improved classification of splice junctions

We use SpliceVec as feature in MLP classifier having one hidden layer. Table 4.4 shows the performance of SpliceVec-g and SpliceVec-sp obtained from the optimal sequence length (10nt flanking + intron) reported in previous subsection. For SpliceVec-g, our classifier yields an accuracy 2.42-17.13% more than that of the alternative approaches whereas for SpliceVec-sp, our classifier outperforms the counterparts by 4.15-18.86%. SpliceMachine predicts a splice site (either donor or acceptor) given a sequence. Since our approach predicts a junction pair (both acceptor and donor), we therefore calculate the performance of SpliceMachine for junction pair by considering a junction pair as true if both the donor and acceptor sites are predicted as true.

Model	Performance measures
	$Ac, Pr, Re, F1(\%)$
SpliceMachine (junction pair)	81.02, 51.59, 85.41, 64.33
DeepSplice	95.73, 95.87, 95.60, 95.74
SpliceVec-g - MLP	98.15, 98.40, 97.89, 98.14
SpliceVec-sp - MLP	99.88, 99.81, 99.95, 99.88

Table 4.4: Performance comparison of different models for splice junction prediction. Performance of SpliceVec is obtained by considering optimal sequence length of 10nt flanking region including complete intronic sequence.

We were also curious to access the performance of SpliceVec for prediction of only donor or acceptor sites. For this, we generate SpliceVec for the consensus dimer (GT for donor and

AG for acceptor respectively) along with 10 nt upstream and downstream sequence. It is then classified with SpliceVec-MLP. The results obtained for both SpliceVec-g and SpliceVec-sp for classification of donor and acceptor sites and its comparison with SpliceMachine is shown in Table 4.5.

Model	Performance measures
	$Ac, Pr, Re, F1(\%)$
SpliceMachine (donor)	92.65, 91.36, 94.20, 92.75
SpliceVec-g - MLP (donor)	84.53, 82.92, 87.02, 84.90
SpliceVec-sp - MLP (donor)	99.86, 99.82, 99.90, 99.86
SpliceMachine (acceptor)	87.01, 85.57, 89.04, 87.27
SpliceVec-g - MLP (acceptor)	80.16, 78.61, 82.89, 80.69
SpliceVec-sp - MLP (acceptor)	99.84, 99.79, 99.89, 99.84

Table 4.5: Performance comparison of SpliceVec-MLP with SpliceMachine for prediction of donor and acceptor sites individually.

We observe that SpliceVec-sp performs better than SpliceVec-g. This is because SpliceVec-g is generated by computing average of vector representations of all the 3-mers in the sequence. Information regarding the order of the 3-mers is not captured in that case. Whereas, SpliceVec-sp captures the ordering information better because it generates the vector representation of a 3-mer based on its neighboring 3-mers in the sequence.

4.3.4 Robust classification of SpliceVec-MLP

We test the robustness of SpliceVec-MLP by analyzing its performance with reduced training examples as well as imbalanced training examples. We vary the number of input samples for training the classifier and observe that SpliceVec-MLP maintains an accuracy more than 98% upto 50% reduction of training samples for SpliceVec-g, whereas for SpliceVec-sp, 99% accuracy is maintained upto 60% reduction of the training samples. On the other hand, the accuracy of DeepSplice reduces by about 6% (Table 4.6). Observing the accuracy of SpliceVec models with reduced dataset, we can conclude that SpliceVec captures more meaningful features compared to DeepSplice.

Train data (%)	DeepSplice	SpliceVec-g	SpliceVec-sp
	$Ac, Pr, Re, F1(\%)$	$Ac, Pr, Re, F1(\%)$	$Ac, Pr, Re, F1(\%)$
100	95.73, 95.87, 95.60, 95.74	98.15, 98.40, 97.89, 98.04	99.88, 99.81, 99.95, 99.88
70	93.90, 94.80, 92.90, 93.84	98.05, 97.77, 98.28, 98.05	99.94, 99.91, 99.98, 99.94
60	93.40, 93.65, 93.11, 93.38	98.02, 98.28, 97.74, 98.02	99.67, 99.96, 99.38, 99.67
50	90.26, 90.24, 90.27, 90.26	98.05, 98.43, 97.65, 98.04	99.62, 99.28, 99.97, 99.63
40	89.79, 88.95, 90.87, 89.90	97.98, 97.42, 98.57, 97.99	99.88, 99.93, 99.83, 99.88

Table 4.6: Classification by reducing training dataset: Performance comparison of DeepSplice and SpliceVec.

To analyze the performance of SpliceVec on imbalanced classes, we vary the ratio of negative to positive samples from 5 to 17 in an interval of 3. This analysis is particularly important because in real scenario the number of negative samples is much more than that of positive samples in a genome. Figure 4.4 shows that the performance of both SpliceVec-g and SpliceVec-sp is significantly more consistent with increasing ratio of negative samples as compared to DeepSplice.

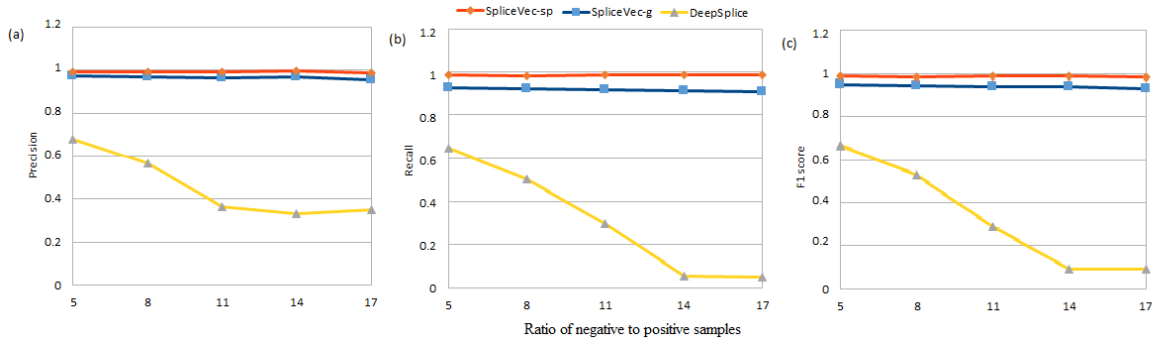


Figure 4.4: Performance of SpliceVec with increasing ratio of negative samples to positive samples. (a) Precision (b) Recall (c) F1-score of SpliceVec-sp, SpliceVec-g and DeepSplice on varying ratio of negative samples from 5 to 17. Data here is positive and negative samples from chromosome 20 of GENCODE dataset.

4.3.5 Prediction performance

The proposed models perform better than state-of-the-art model in detecting non-canonical true splice junctions. SpliceVec-g identifies upto 192 (84.95%) out of 226 non-canonical true splice junctions present in our test dataset. SpliceVec-sp identifies all the 226 (100%) non-canonical splice junctions. The capability of the classifier to detect non-canonical splice junctions significantly improves on including complete intron sequence for generating SpliceVec.

Observing 100% accuracy in detecting non-canonical splice junctions, we further modify our decoy splice junction samples by including non-consensus donor and acceptor dimers

(dimers other than GT-AG pair) in it. In order to include non-consensus dimer-pairs, we consider two common classes of exceptions to consensus splice site sequences reported in [93]. We consider 0.5% AT-AC dimer-pairs and 0.5% GC-AG dimer-pairs based on their reported frequency of occurrence. In this scenario, SpliceVec-g identifies 188 (83.18%) out of 226 non-canonical true splice junctions whereas SpliceVec-sp identifies all the 226 (100%) junctions in the worst case out of 5 simulations (considering 10 nt flanking region + intron). Performance of SpliceVec-MLP indicates that the feature representation by SpliceVec is invariant to canonical and non-canonical splice junctions.

Model	SpliceVec-g		SpliceVec-sp	
	$Pred_{can}$	$Pred_{non-can}$	$Pred_{can}$	$Pred_{non-can}$
10nt flanking	72,446	51	87,635	222
20nt flanking	74,172	84	87,375	223
30nt flanking	75,222	127	87,367	222
40nt flanking	77,794	141	87,098	223
10nt flanking + intron	86,037	190	87,854	226
20nt flanking + intron	85,656	192	87,884	226
30nt flanking + intron	84,814	191	87,888	226
40nt flanking + intron	84,190	186	87,899	225

Table 4.7: Correctly predicted canonical ($Pred_{can}$) and non-canonical ($Pred_{non-can}$) positive splice junctions, out of 87,927 canonical and 226 non-canonical junctions, using MLP and both SpliceVec-g and SpliceVec-sp by varying the length of junction sequence.

We also observe that the most common correctly identified dinucleotide sequence at the donor site of non-canonical splice junction is AT whereas that at the acceptor site is AC, constituting about 43% of the non-canonical splice junctions. This is in consistency with experimentally studied annotation which identifies"AT-AC" introns as one of the most important classes of exception to splice site consensus [93]. Table 4.7 shows the prediction of true splice junctions in the worst case out of 5 simulations on varying the sequence length for both SpliceVec-g and SpliceVec-sp. Out of the 87,927 canonical and 226 non-canonical true splice junctions, DeepSplice identifies 84,170 (95.73%) canonical and 97 (42.92%) non-canonical splice junctions. Thus, SpliceVec-MLP shows 2.12% (4.23%) higher performance than DeepSplice in terms of identification of canonical splice junctions using SpliceVec-g (SpliceVec-sp). In terms of identification of non-canonical splice junctions, there is a 42.03% (57.08%) improvement in performance by SpliceVec-g (SpliceVec-sp) compared to DeepSplice.

We compare the time taken for classifying the test samples. SpliceVec-MLP classifies 72,268 test samples per second of CPU time in the worst case out of 5 simulations. DeepSplice, on the other hand, classifies 5,585 test samples per second. SpliceVec-MLP is therefore 12.94 times faster than DeepSplice. With SpliceVec-sp, the classifier can identify all the splice junctions belonging to some of the important genes like *TSLP* and *BATF2* that

are known to be involved in several diseases. *TSLP* causes diseases like atopic dermatitis, eosinophilic esophagitis, and allergic rhinitis [30]. *BATF2* has been known to be associated with cancer and some allergic diseases [46].

4.4 Chapter Summary

We design learning models that identify true splice sites based on sequence embedding where features are learnt by the model on its own. This work establishes that a robust model based on only sequence-based features can be designed to predict splice sites, specially performing remarkably well with non-canonical splice sites. It studies the sequences at word and sentence levels and compares the result. However, motif discovery is difficult in this type of representation models due to limitations in intuitively explaining the embedding space. Therefore, in the next contribution, we aim at applying neural model that can identify splice sites by learning splicing features by itself as well as extract and interpret the biologically relevant features learnt by the model.

"Research is something that everyone can do, and everyone ought to do. It is simply collecting information and thinking systematically about it."

Raewyn Connell
Australian sociologist

5

SpliceVisuL: Visualization of Bidirectional Long Short-term Memory Networks for Splice Junction Prediction

5.1 Introduction

This chapter is based on the second objective of the thesis. This work also addresses the limitation of the previous contribution where we could not extract the features learnt by the model. In this work, we apply a neural network model that can identify splice sites by learning relevant splicing features by itself from the genome sequences. Additionally, the biologically significant features learnt by the model can also be extracted. Several visualization techniques have been effectively applied across various domains to analyze learning models and infer relevant features. Our work aims to apply suitable visualization techniques to identify the features that contribute to the prediction performance. Towards this aim, we also ask the following questions-

Question 1: How can various visualization techniques be adapted for identifying the relevant features for a particular task?

Question 2: Do all visualization techniques deliver similar results, or is one method superior to the others?

We employ five different visualization techniques by modulating them to suit the genome sequences as input. Based on the time when visualization is obtained, we can group the chosen visualization techniques into two categories, namely intrinsic visualization and post-hoc visualization [36]. We achieve the intrinsic visualization by adding an interpretable component, namely *attention layer*, to the model. This visualization technique identifies features considering all the sequences present in the dataset. The post-hoc visualization techniques employed can be further categorized into back-propagation based techniques and perturbation based techniques [36]. This set identifies motifs based on individual sequences. The post-hoc visualization techniques used are *smooth gradients of noisy nucleotide embeddings*, *integrated gradients of nucleotide embeddings*, *omission of a single nucleotide*, and *occlusion of k-mers*. Among these visualization techniques, the first two are back-propagation based,

whereas the next two are perturbation based.

We evaluate the different visualization techniques in their ability to infer relevant features known to be important for splice junction identification. Most of the existing computational methods focus only on the identification of canonical splice junctions due to the lack of consistent non-canonical consensus. Nevertheless, the non-canonical splicing signals are equally important in understanding the splicing phenomenon [95], and hence this remains an interesting area to be investigated further.

Several studies [38, 74, 75, 139] have applied neural network-based models to identify canonical and non-canonical splice junctions but with limited or no inference of learnt information. Learning models used in [142], [138], and [59] consider only the canonical splice junctions. Also, these works apply one chosen visualization technique for the inference of learnt features. These studies do not compare the performance of various visualization methods. This work fills the research gaps mentioned above by application and comparison of various visualization methods on the extraction of known canonical and non-canonical features.

Lanchantin et al. [69] explore various sequence-specific as well as class-specific visualization techniques to obtain the important nucleotide positions present in a genome sequence for classification of transcription factor binding sites. In contrast to their work, we have incorporated variable length occlusion to study variable length canonical as well as non-canonical splicing features apart from accessing the importance of each nucleotide position using all the visualization techniques. We also apply the smooth gradients of noisy nucleotide embeddings, rather than the raw gradients, to generate sharper *sensitivity maps* [12].

Motivated by application of recurrent neural network (RNN) in sequence-based bioinformatics problems [49, 73, 74], we further explore its application in splice junction prediction by employing bidirectional long short-term memory (BLSTM) networks [45]. There have been earlier attempts at applying RNN variants like long short-term memory (LSTM) and gated recurrent unit (GRU) on this task [74]. We further superimpose the model with an attention [13] layer to add interpretability to the model. The contributions of this chapter can be summarized as follows:

- We explore the application of BLSTM network with attention for the prediction of splice junctions. The proposed architecture achieves state-of-the-art performance.
- We generate two different types of negative dataset to test the consistency of RNN models compared to other neural network models.
- We redesign some of the effective visualization techniques, available in the literature, to be capable of comprehending genome sequences as inputs.
- We infer relevant biological information learnt by the model for both canonical and non-canonical splicing events. The splicing features are validated with the existing knowledge from the literature.

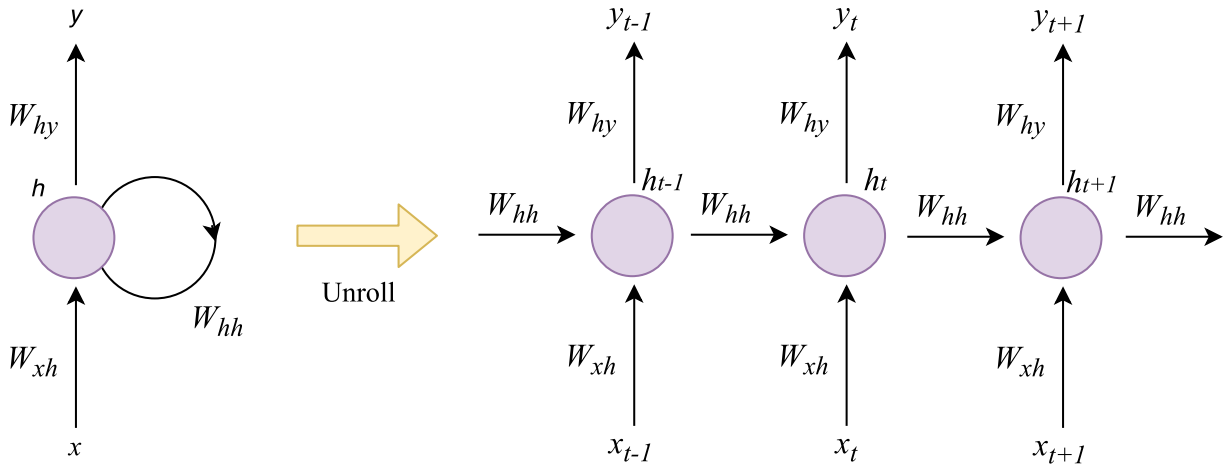


Figure 5.1: An unrolled RNN.

- We further compare and discuss the ability of the visualization techniques in the inference of various known canonical and non-canonical features.
- We provide the source code of the proposed prediction tool, named SpliceVisuL, for prediction and visualization of splice junctions.

5.2 Methods

In this section, we introduce the neural architecture employed for the classification of true and decoy splice junctions. We discuss the preliminaries required for understanding the neural network employed here. Further, we discuss the visualization techniques applied to analyse the features learnt by the model.

5.2.1 Preliminaries on model architecture

5.2.1.1 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 5.1, 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:

$$y_t = W_{hy}h_t$$

5.2.1.2 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 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 5.2 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:

$$h_t = o_ttanh(C_t)$$

5.2.1.3 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 5.3 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

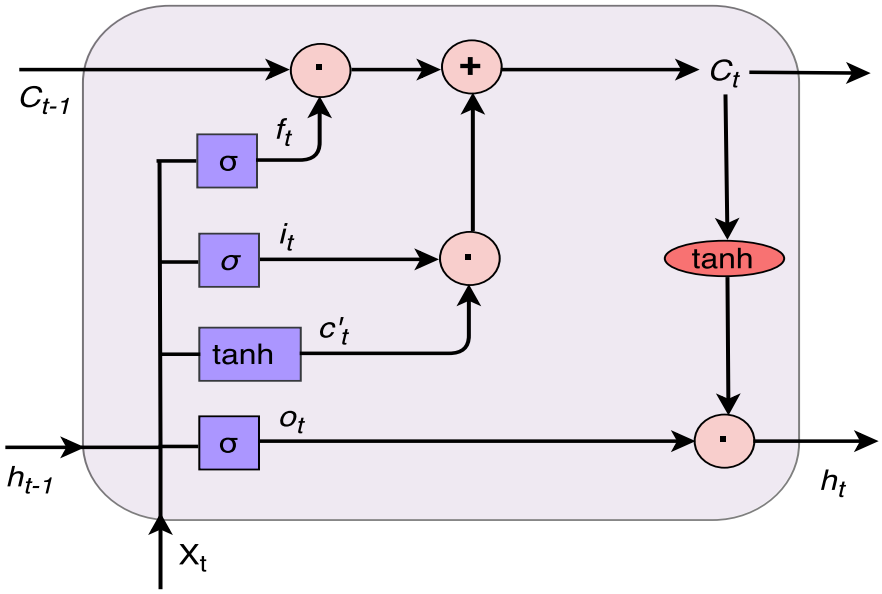


Figure 5.2: An LSTM cell.

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:

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

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

$$y_t = f(W_{\overrightarrow{h}y} \overrightarrow{h}_t + W_{\overleftarrow{h}y} \overleftarrow{h}_t)$$

5.2.1.4 Attention mechanism

Attention based neural networks are successful in diverse tasks like speech recognition [29] and language translation [13], 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 [13]. 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

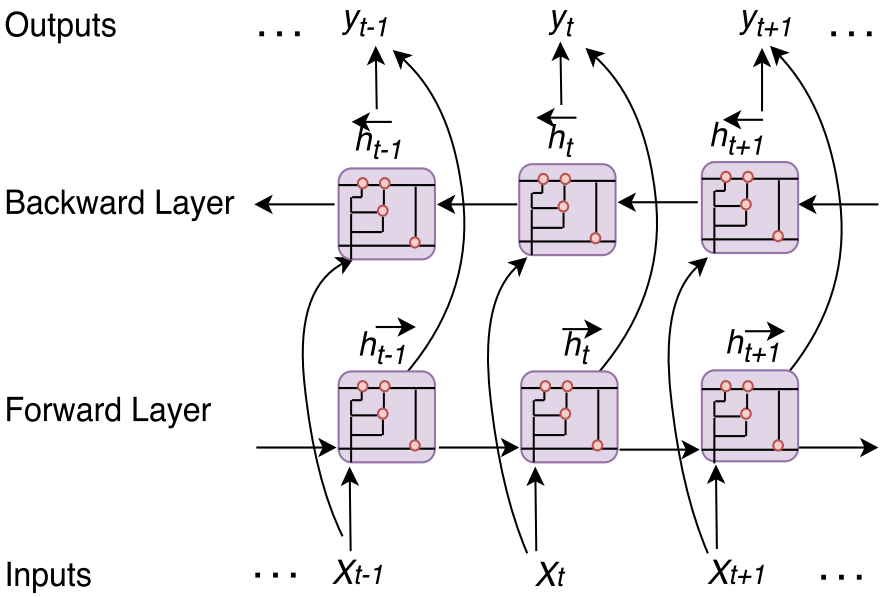


Figure 5.3: A BLSTM network.

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

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.

5.2.2 Neural architecture

The overview of *SpliceVisuL* architecture is shown in Figure 5.4. We discuss the entire workflow in the following subsection.

5.2.2.1 Input representation

Input is a putative splice junction sequence consisting of five nucleotide codes, *A (Adenine), C (Cytosine), G (Guanine), T (Thymine), and N (any one of the four nucleotides).* We use a dense vector representation for each nucleotide code. Each input sequence is passed through an embedding layer, which transforms each input splice junction sequence of length n into an $n \times 4$ dimensional dense vector that gets updated while training the neural network.

5.2.2.2 Modeling splice junctions using BLSTM network

The $n \times 4$ dimensional dense input vectors are fed in mini-batches into both forward and backward LSTM [41] layers configured as a BLSTM. Both the LSTM layers learn meaning-

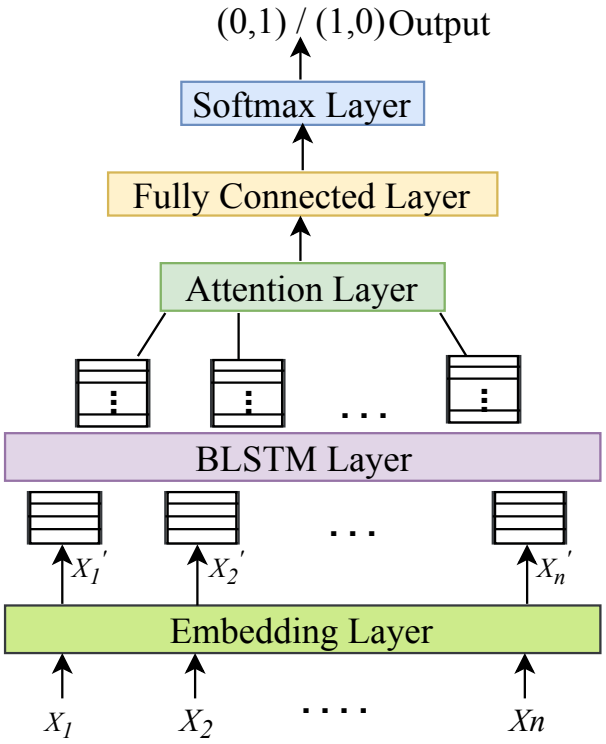


Figure 5.4: An overview of SpliceVisuL architecture.

ful features in a supervised manner to generate an $n \times n_l$ dimensional vector representation, where n_l is the number of hidden units in each LSTM layer. Both the vectors, generated by the forward and backward LSTM layers, are concatenated to generate an $n \times 2n_l$ dimensional vector representing the learnt features of each splice junction.

5.2.2.3 Feature interpretation using attention layer

The attention layer is added to capture the role each nucleotide in the input sequence plays in the classification of the splice junction. This layer adds the ability of intrinsic visualization to the model. We implement the traditional attention mechanism proposed in [13]. The $n \times 2n_l$ representation of each sequence obtained from the BLSTM network is fed into the attention layer. The attention weights are fed into a fully connected layer and eventually to a softmax layer to obtain the classification results. We use binary cross-entropy and Adam [65] as the loss function and the optimizer, respectively.

5.2.3 Visualization techniques

The visualization techniques rely on measuring the change in the performance of the trained model due to a change in the input sequence. The change can be implemented at a single nucleotide or a span of consecutive nucleotides. These visualization techniques add post-hoc

interpretability to the model. The post-hoc visualization techniques used in this work can be categorized as perturbation based and back-propagation based visualizations.

Perturbation based visualization techniques used are based on the modification/masking of the input sequence. Back-propagation based techniques used are based on the modification of the embedding space of the model. In some sense, the perturbation based techniques resemble the site-specific mutagenesis [68] performed in a wet lab setup. The visualization techniques are applied to define a scoring function, referred to as the *deviation value*. The *deviation value* of a genomic region reflects the contribution of that region to the classification score. We evaluate the visualization techniques for the inference of known splicing features, based on these *deviation values*. The following subsections describe the various visualization techniques.

5.2.3.1 Smooth gradients of noisy nucleotide embeddings

In image classification tasks, the gradient of the unnormalized output probabilities with respect to the input image (referred to as *sensitivity maps* [12]), indicates how much a tiny change in a pixel can affect the final output. Thus, to incur a minute change in the input sequence, we add noise to the embeddings of the nucleotides and compute the change in classification score. However, the sensitivity maps, resulting from raw gradients, are usually noisy [119]. Therefore, based on the concept of smooth gradient [119], we average out the gradients obtained from several different noisy embeddings for each position of a sequence. For each input sequence, we generate 50 samples of embeddings by adding Gaussian noise with a standard deviation of 0.15. The average gradient at each sequence position, named *smooth gradient*, is the *deviation value* in this case. As a result, one might expect that the resulting averaged sensitivity map would crisply highlight the key regions.

5.2.3.2 Integrated gradients of nucleotide embeddings

Integrated gradients is a back-propagation based visualization technique proposed by Sundararajan et al.[122]. Similar to the smooth gradients of noisy nucleotide embeddings, the integrated gradient is also based on the computation of sensitivity maps. However, instead of computing a single gradient of the output probability with respect to the input, the integrated gradient computes the average gradient as the input varies stepwise along a linear path from a baseline to the given input [122]. We consider 50 steps for our experiments. The average gradient at each sequence position is the *deviation value* here. The baseline in our case is the zero embedding vector, as suggested in [122] for text inputs.

The reason for the selection of integrated gradients over other back-propagation based visualization techniques like basic gradients, DeepLIFT [117], and Layer-wise relevance propagation (LRP) [19] can be explained by the following factors. The integrated gradient satisfies two desirable properties of attribution methods: sensitivity and implementation invariance, either of which is not satisfied by basic gradients, DeepLIFT and LRP.[122]

An attribution method is said to satisfy the sensitivity property when for any input-baseline pair, which differs in one feature and has different predictions, the differing feature is assigned a non-zero attribution. This property is desired in any attribution method because the lack of this property intuitively means that the attribution method may focus on irrelevant features. Gradient based methods lack this property, whereas methods like DeepLIFT and LRP satisfy it.[122]

However, DeepLIFT and LRP lack the property of implementation invariance. An attribution method is considered invariant to the implementation model when it assigns identical feature attributions for functionally equivalent models. Lack of this property may lead to the attribution methods being sensitive to the unimportant biases of the implementation model.[122] Also, in models like RNN with multiplicative interactions (like LSTM and BLSTM), DeepLIFT fails to produce meaningful results.[7] Although there are researches inferring that a more principled backpropagation rule can produce improved results for DeepLIFT in the case of multiplicative units.[83, 103]

5.2.3.3 Omission of a single nucleotide

The feature vector, obtained from the fully connected layer of *SpliceVisuL*, represents the complete input sequence. To measure the significance of each sequence position, we calculate its *omission score* [60]. The omission score of the j^{th} position p_j^i in a sequence s_i is given by

$$omission(p_j^i, s_i) = 1 - cosine(V_{s_i}, V_{s_i \setminus p_j^i}) \tag{5.1}$$

where V_{s_i} is the feature representation obtained from the fully connected layer for the sequence s_i, and $V_{s_i \setminus p_j^i}$ is the feature representation obtained from the same layer for the same sequence with the nucleotide at position p_j^i replaced by N. $Cosine(V_{s_i}, V_{s_i \setminus p_j^i})$ measures the similarity of the two vectors and is calculated as

$$cosine(V_{s_i}, V_{s_i \setminus p_j^i}) = \frac{V_{s_i} \cdot V_{s_i \setminus p_j^i}}{\|V_{s_i}\| \|V_{s_i \setminus p_j^i}\|} \tag{5.2}$$

Therefore, the omission score measures the *deviation value* of the vector representations, with and without the omitted nucleotide. A higher deviation implies a higher significance of that sequence position.

5.2.3.4 Occlusion of k-mers

We occluded portions of a sequence to observe the variation in the predicted output. This approach has its motivation from [134]. For each sequence, we run a sliding window w_l of

length l, centered at nucleotide number $(l+1)/2$, and replace the nucleotides within the window with N. We pass the modified sequence through the model to obtain the corresponding *deviation value*. The *deviation value* is given by the absolute difference of model outputs with and without occlusion. For occlusion of a window of length l, the *deviation value* is stored in the center, that is, in position $(l+1)/2$, of the window.

We generate *deviation values* for test sequences in batches. For a batch of size B, the *deviation values* are in the form of a matrix of size $B \times n$, where each input sequence is of length n. 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 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*. We propose two variations of occlusion described as follows:

Fixed length occlusion: This considers occlusion of k nucleotides (denoted by *occlusion-k*), where k is 1 or 3 in our experiments. *Deviation values* at boundary indices are computed by occluding the first and the last $(k+1)/2$ indices. The significance of a genomic region is proportional to the corresponding *deviation value*.

Variable length occlusion: Fixed length occlusion has a limitation of considering fixed length genomic regions, whereas in real scenario there may be sequence patterns of variable lengths that regulate splicing. Hence, we incorporate variable length occlusion, where for each index j of a sequence s_i, we occlude a window w_l of length $l \in \{1, 3, 5, 7, 9, 11\}$. We compute the *deviation values* denoted by $dev_l^{i,j}$, for each window length l, with and without occlusion. The *deviation value* assigned to position $(l+1)/2$ of window w_l is given by $max_l(dev_l^{i,j})$ for $l \in \{1, 3, 5, 7, 9, 11\}$. The window length $argmax_l(dev_l^{i,j})$ corresponding to index j of sequence s_i is stored in the j^{th} column of the i^{th} row of a *window matrix*. Therefore, the value in j^{th} column of i^{th} row of the *window matrix* signifies the length of the pattern, centered at index j of sequence s_i, that contributes maximum to the prediction of the model.

5.2.4 Prerequisites for visualization

For each of the visualization techniques, we extract the relevant features learnt by the model using the following methods. We extract the features separately over the sets of canonical and non-canonical test sequences. Henceforth, the splice junction sequences comprising the canonical dimer motifs ($GT - AG$) are referred to as the canonical sequences, and the remaining sequences are referred to as the non-canonical sequences. The first three methods are adapted from [142] for analysing the same splicing features as discussed in [142], but considering both canonical and non-canonical sequences.

1. Average deviation value per position: At each sequence position, we compute the average absolute deviation values across all the sequences irrespective of the nucleotide type.

2. Average deviation value per position per nucleotide: At each sequence position, we compute the average deviation values per nucleotide across all the sequences.

3. Average deviation value of a specific pattern in a specific region of the sequence: We compute the deviation value of a specific pattern by summing up the deviation values of each nucleotide in the pattern. For each starting position in the specific region, we calculate the average deviation values of all occurrences of the pattern across the sequences.

4. Frequency of different occlusion windows per position: We compute the frequency of different window lengths per position based on the number of times occlusion of that window length centered at that position contributed to maximum deviation values across all sequences. In other words, we compute the frequency of window lengths per position based on the *window matrix*, explained in Section 5.2.3.4.

5.3 Experimental setup

5.3.1 Positive data generation

We use GENCODE annotations [47], based on human genome assembly version *GRCh38*, to generate the dataset. We target to assess the model's performance on the prediction of novel splice junctions. Several alignment-based methods [11, 124] have the potential to identify novel splice junctions through *ab initio* alignment of RNA-seq reads to the reference genome. The machine learning-based approach [138, 139] was also applied recently for the identification of novel splice junctions. To this end, we train the model using an earlier release (GENCODE annotation version 20) and test the model on only the newly added splice junctions in a later release (GENCODE annotation version 26), as described in [139].

Unlike some contemporary state-of-the-art models like [75] and [142] that consider either acceptor or donor splice sites as the input data, we consider junction pairs as the input for training and testing the model. Junction pairs were also chosen as input dataset in other works [38, 138, 139]. Training the model with junction pairs results in performance improvement of the model, as shown in Table 5.1.

We compare the performances of various models in the prediction of only a donor junction with flanking regions as well as a junction pair with flanking regions. We observe that all the models perform better in prediction of junction pairs compared to prediction of donor junctions alone. The improvement in the model's performance can be justified by the study which suggests that splice sites are not recognized independently through individual consensus but usually in pairs across exons or introns [18, 59].

Table 5.1: Performance of the various models in donor splice junction and junction pair prediction. Accuracy (Ac), Precision (Pr), Recall (Re) and F1 Score (F1) are computed in percentage.

Model	Donor prediction	Splice junction prediction
	$Ac, Pr, Re, F1(\%)$	$Ac, Pr, Re, F1(\%)$
SpliceVisuL	92.98, 96.19, 89.52, 92.74	98.38, 99.38, 97.38, 98.37
DeepSplice	90.67, 91.10, 90.15, 90.62	92.85, 96.43, 89.03, 92.56
SpliceRover	88.68, 87.35, 90.61, 88.90	90.13, 89.90, 90.41, 90.16
SpliceVec-MLP	74.19, 71.31, 80.95, 75.82	93.98, 96.43, 91.40, 93.78

We extract 291,831 and 294,576 unique introns from version 20 and version 26 of the GENCODE annotations, respectively. The introns are extracted from protein-coding genes only. Each intron is extracted with a flanking upstream and downstream region at donor and acceptor splice junction, respectively. We observe that the length of the extracted introns varies in the range from 1 nucleotide (nt) to 1,240,200 nt. From each intron, we truncate donor and acceptor splice junctions with an equal upstream and downstream flanking region. We obtain a splice junction pair by concatenating the truncated donor and acceptor junctions of an intron. We consider introns of length greater than 30 base pairs (bp). This choice is based on a study [102] which states that introns shorter than 30 bp usually result from sequencing errors in the genome [38]. This reduces the number of junction pairs to 290,502 and 293,889 in versions 20 and 26, respectively. Our test data comprises 5,612 novel junction pairs present only in version 26.

5.3.2 Negative data generation

Existing works usually adopt one of the following two techniques to generate the negative data. We adopt both the ways of generating the negative data to have comprehensive and unbiased analysis. The Type-1 dataset is generated similar to the standard procedure described in [99]. We extract a portion of the sequence from the center of each intron to generate a pseudo sequence. The length of the extracted portion is kept equal to the length of an input sequence. This set of negative data captures the non-randomness of DNA sequences. We obtain 290,502 false samples for training data and 5,612 false samples for testing data using this procedure.

More than 98% of splice junctions contain the consensus dimers GT and AG at the donor and the acceptor junctions, respectively [24]. However, the occurrence of the consensus dinucleotide is far more frequent compared to the number of true splice junctions in the genome. The neat exclusion of the pseudo sites, by the splicing mechanism, suggests the presence of other subtle splicing signals which play an important role in the process. The

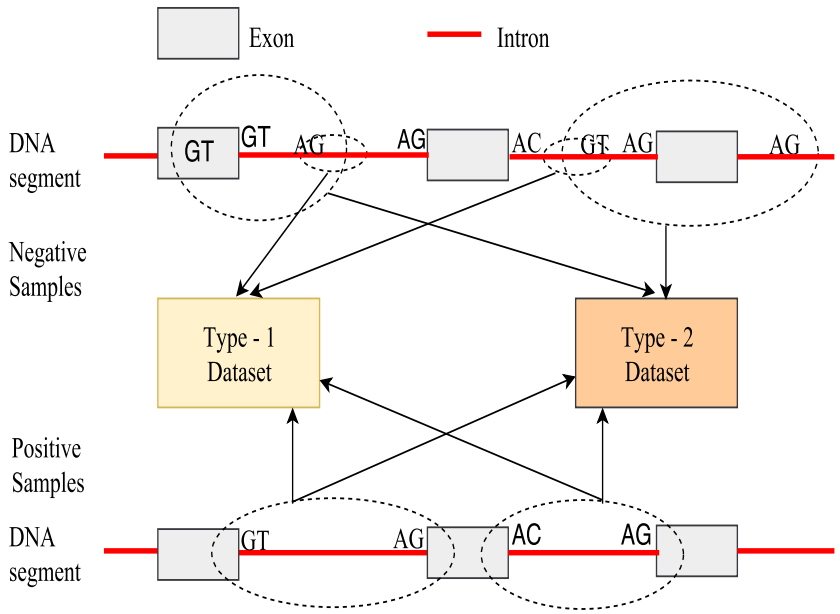


Figure 5.5: A pictorial representation of the Type-1 and Type-2 dataset. For simplicity of representation, the entire extracted DNA region for the positive samples and the Type-2 negative samples are displayed using the dashed circles. These samples are truncated to 40 nt upstream and downstream regions before feeding into the learning model.

presence of consensus dimer in all the negative samples will result in the model learning the remaining splicing patterns present in the vicinity of the splice junctions.

Therefore, we generate the Type-2 dataset based on the procedure described in [38, 139], where the negative data is randomly sampled from the human genome assembly version *GRCh38*. For each decoy junction pair, we randomly search for the consensus dimers *GT* and *AG* such that both lie in the same chromosome, and the distance between them lies in the range of 30 nt and 1,240,200 nt. We obtain a huge number of such samples using this procedure, out of which we randomly select 290,502 false samples for training data and 5,612 false samples for testing data. Both the scenarios are pictorially depicted in Figure 5.5. The preprocessed dataset can be downloaded from `https://www.iitg.ac.in/anand.ashish/resources.html`.

5.3.3 Training and Hyperparameter tuning

Each input splice junction is truncated to 40 nt upstream and downstream flanking regions of the consensus dimer *GT* or *AG*, thus obtaining an 82 nt sequence. Both the donor and the acceptor junctions of a junction pair are concatenated to form a 164 nt sequence. The effect of variation in the flanking region on model accuracy is shown in Section 5.3.4.

SpliceVisuL can be represented as a (1-4-100-100-2048-2) architecture, where 4-dimensional embeddings are passed through a BLSTM, attention, and fully connected layer with 100,

Table 5.2: 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

100, and 2048 units, respectively. Values for batch size, dropout, recurrent dropout, and epochs are set to 128, 0.5, 0.2, and 50, respectively. The hyperparameters are tuned by partitioning the training data from version 20 into 90% training and 10% validation data. All experiments were carried out on an NVIDIA GeForce GTX 980 Ti GPU machine with 6GB memory. We evaluate the performance of the classifier based on precision, recall, accuracy, and F1 score.

5.3.4 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 chapter are performed with a flanking region of 40 nt. The performance of the model with varying flanking region is shown in Table 5.2.

5.3.5 Variation in model architecture

Table 5.3 shows the variation in the performance of the model with variation in the number of hidden layers (HL). Results are shown for Type-2 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 chapter are performed with a model having one hidden layer of BLSTM units.

Table 5.3: Performance of the model with variation in architecture

Model	Performance measures
	$Ac, Pr, Re, F1(\%)$
LSTM with 1 HL	97.68, 98.88, 96.45, 97.65
LSTM with 2 HL	98.49, 99.10, 97.88, 98.48
BLSTM with 1 HL	98.38, 99.38, 97.38, 98.37
BLSTM with 2 HL	98.24, 99.66, 96.81, 98.21

5.3.6 Baselines

We implemented the following state-of-the-art models as baselines and compared the results obtained by the various models on the same set of training and testing data. The hyperparameters for the baselines were tuned using the same process mentioned in Section 5.3.3.

1. Vanilla LSTM: This model comprises an embedding layer, two hidden LSTM layers, and a softmax output layer, as proposed in [74].
2. LSTM with attention: We replaced the hidden units of the proposed architecture with LSTM units.
3. SpliceRover: This model classifies an input sequence as a donor (acceptor) or not a donor (acceptor) in the donor (acceptor) classification model. The classifier comprises multiple alternating convolutional and max-pooling layers [142].
4. DeepSplice: This model classifies input sequences using a deep CNN composed of two convolutional layers [139]. Input sequences are represented in the form of a $4 \times N$ matrix comprising the flanking 'N' nucleotides in the upstream and downstream of both acceptor and donor splice junctions.
5. SpliceVec-MLP: This model learns feature vectors of the entire intronic region, along with flanking upstream and downstream exonic regions, to be classified using an MLP [38]. The input formation for SpliceVec-MLP is described in Section 5.3.7.

5.3.7 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.

5.3.8 Hyperparameters of the various baselines

Discussed below are the hyperparameters tuned for 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 [38]: 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 [139]: We consider a batch size of 160, dropout keep probability as 0.8 and the Adam optimizer learning rate as 0.001.
- SpliceRover [142]: We consider a batch size of 64, learning rate as 0.05 with decay rate of 0.5 every 5 steps and Stochastic Gradient Descent with Nesterov momentum 0.9 as the update strategy. We consider only one convolutional and one max-pooling layer as this architecture provided the best performance.

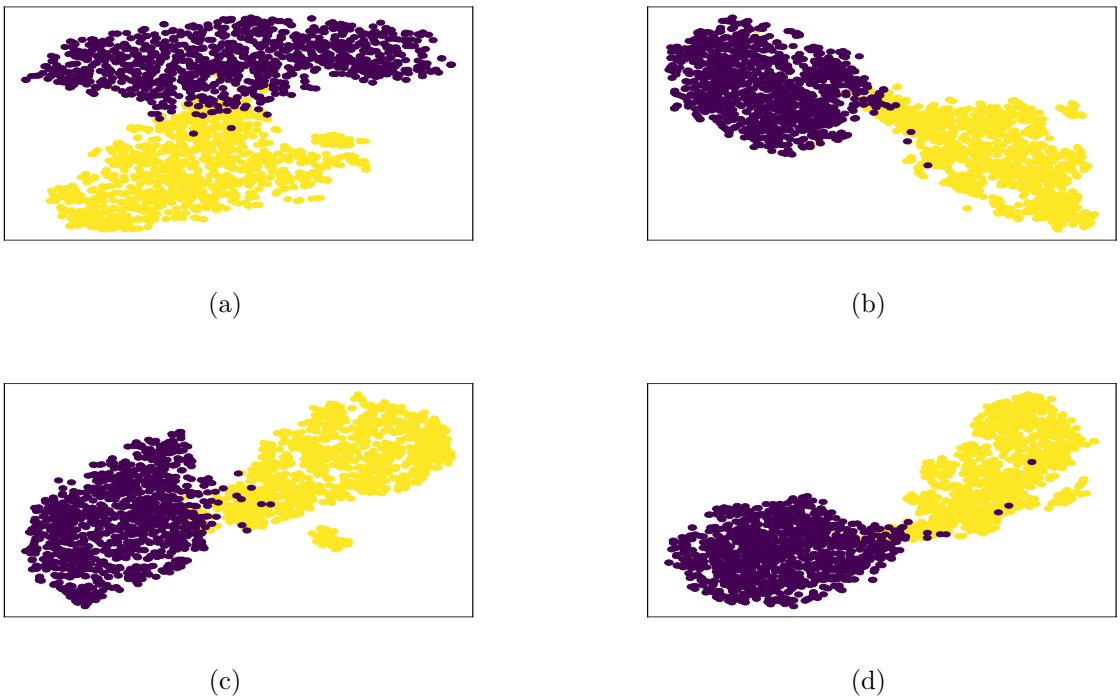
5.4 Results

5.4.1 Prediction performance

To assess the quality of feature embedding, obtained from the dense layer of *SpliceVisuL*, we plot the 2048-dimensional vector by reducing it to a two-dimensional vector using Stochastic Neighbor Embedding (t-SNE) [84]. We observe that the projected representations are distinctly separable in the two-dimensional feature space. Figure 5.6 shows t-SNE plots of 4 different randomly selected sets of 1000 true and 1000 decoy splice junctions [1].

We evaluate the performance of *SpliceVisuL* using both Type-1 and Type-2 dataset. The evaluation is done in terms of accuracy, precision, recall, and F1-score. We compare the predictive performance with the baselines, described in Section 5.3.6. Table 5.4 shows comparison of *SpliceVisuL* with the baselines. We observe a performance improvement of the RNN based architectures, over other neural network-based baselines, in the range of 2%-27% for the Type-1 dataset and an improvement in the range of 4%-8% for the Type-2 dataset. The RNN based models are consistent in their performance with both the dataset.

[1] We plotted 1000 junctions considering the execution time required for larger sample sizes.



(a) (b) (c) (d)

Figure 5.6: t-SNE plots of 4 random sets of 1000 true and 1000 decoy splice junctions. Points in blue represent true, whereas points in yellow represent decoy splice junction pairs.

We analyse the performance of *SpliceVisuL* on the Type-2 dataset.

5.4.2 Visualization of splicing features

We apply the various visualization techniques on *SpliceVisuL* to decipher the features learnt by the model. The following analysis has been done on 10853 canonical and 371 non-canonical test sequences.

5.4.2.1 Identifying the significance of sequence positions near splice junctions

All the visualization techniques identify nucleotides in the proximity of splice junctions as the most significant, based on average deviation value per position. The nucleotide importance decreases as we move further in the flanking region. Figures 5.7(a) (5.7(c)) and 5.7(b) (5.7(d)) show the deviation values obtained from occlusion-1 (smooth gradients) at canonical donor and acceptor splice junctions, respectively. Figures 5.7(e) and 5.7(f) show the deviation values obtained from occlusion-1 and smooth gradients at non-canonical acceptor splice junctions, respectively.

We observe relatively higher importance of the intronic region at canonical acceptor

Table 5.4: Performance of *SpliceVisuL* compared with state-of-the-art models. Accuracy (Ac), Precision (Pr), Recall (Re), and F1 Score (F1) are computed in percentage.

Model	Type-1 Dataset	Type-2 Dataset
	$Ac, Pr, Re, F1(\%)$	$Ac, Pr, Re, F1(\%)$
SpliceVisuL	98.24, 99.66, 96.81, 98.21	98.38, 99.38, 97.38, 98.37
LSTM	98.19, 99.45, 96.91, 98.16	98.36, 99.38, 97.34, 98.35
LSTM with attention	98.24, 99.70, 96.77, 98.21	97.68, 98.88, 96.45, 97.65
SpliceRover	96.03, 95.98, 96.10, 96.04	90.13, 89.90, 90.41, 90.16
DeepSplice	89.81, 94.77, 84.43, 89.05	92.85, 96.43, 89.03, 92.56
SpliceVec-MLP	71.57, 71.42, 72.28, 71.72	93.98, 96.43, 91.40, 93.78

splice junction due to the presence of the polypyrimidine tract (PY-tract, see Section 5.4.2.2). Both perturbation based occlusion and back-propagation based smooth gradients assign similar importance to genomic regions in the case of canonical splice junctions.

However, in the case of non-canonical splice junctions, both these techniques show different trends. Smooth gradients assign relatively lower importance to the PY-tract, which is consistent with the knowledge that non-canonical splice junctions have weakly conserved PY-tract [95]. Occlusion, on the other hand, assigns relatively higher importance to the upstream region of the acceptor junction.

The higher importance assigned by occlusion can be attributed to a characteristic of perturbation based visualization techniques explained in [7]. The characteristic illustrates that perturbation based methods are better in explaining the role of individual features in isolation, whereas back-propagation based methods are better in capturing the effect of multiple features together. Considering each nucleotide position as a feature, occlusion assigning higher importance to the upstream region suggests that features in this region work in isolation, rather than as consecutive consensus, in the case of non-canonical splicing. This is inferred in Section 5.4.2.4 also.

5.4.2.2 Identifying the splicing motifs at splice junctions

Canonical motifs: We compare the extracted splicing motifs with the existing consensus of canonical splice junctions known from the literature (Figure 2.3). The donor and acceptor site consensuses also comply with the consensuses obtained from our dataset.

Figures 5.8(a), 5.8(e), 5.8(c), and 5.8(g) show the donor and the acceptor junction motifs captured by occlusion-1 and omission, respectively. Both the visualizations capture most of the known consensus. We also observe that the donor and the acceptor junction motifs

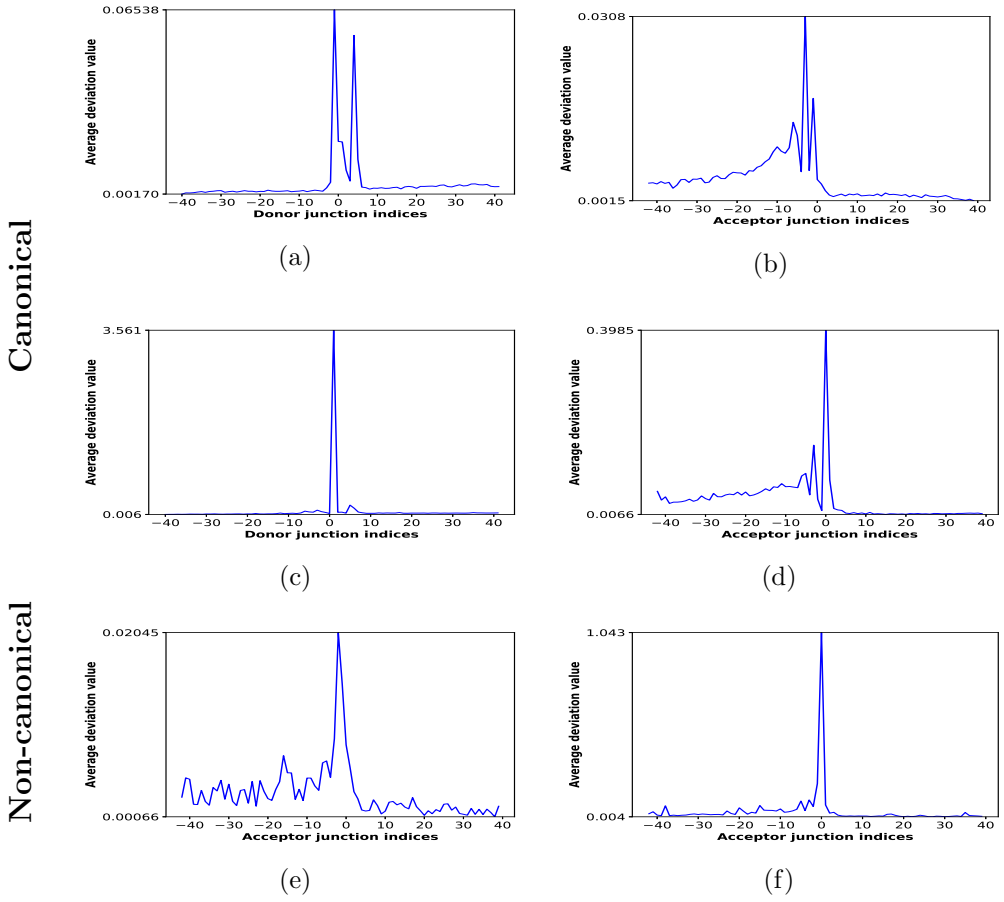


Figure 5.7: Significance of sequence positions near splice junctions. The average deviation value per nucleotide position is shown for occlusion-1 of canonical (a) donor and (b) acceptor splice junctions, smooth gradient of canonical (c) donor and (d) acceptor splice junctions, (e) occlusion-1 of non-canonical acceptor and (f) smooth gradient of non-canonical acceptor splice junctions.

(Figures 5.8(b) and 5.8(f)) captured by occlusion-3 are better than occlusion-1. This is due to the inherent behavior of occlusion which suggests that larger occlusion windows imply better learnt representations [7]. This is also observed in our results in Section 5.4.2.3.

However, back-propagation based visualizations capture motifs partially. Integrated gradients learn better representation for donor junction (Figure 5.8(d)), whereas smooth gradients learn better representation for acceptor junction (Figure 5.8(h)). However, DeepLIFT [117], a back-propagation based visualization technique, identified both donor and acceptor junction consensus in [142]. This can be attributed to the factors that our dataset comprises donor-acceptor junctions pairs as well as both canonical and non-canonical sequences.

Figure 5.9 shows the donor and the acceptor splice junction motifs captured by attention. Attention visualization is limited in a way that it identifies important sequence positions considering the overall dataset. Hence, it does not provide information regarding what class the feature is important for [94]. Therefore, it possibly overlooks relevant signals regulating splicing.

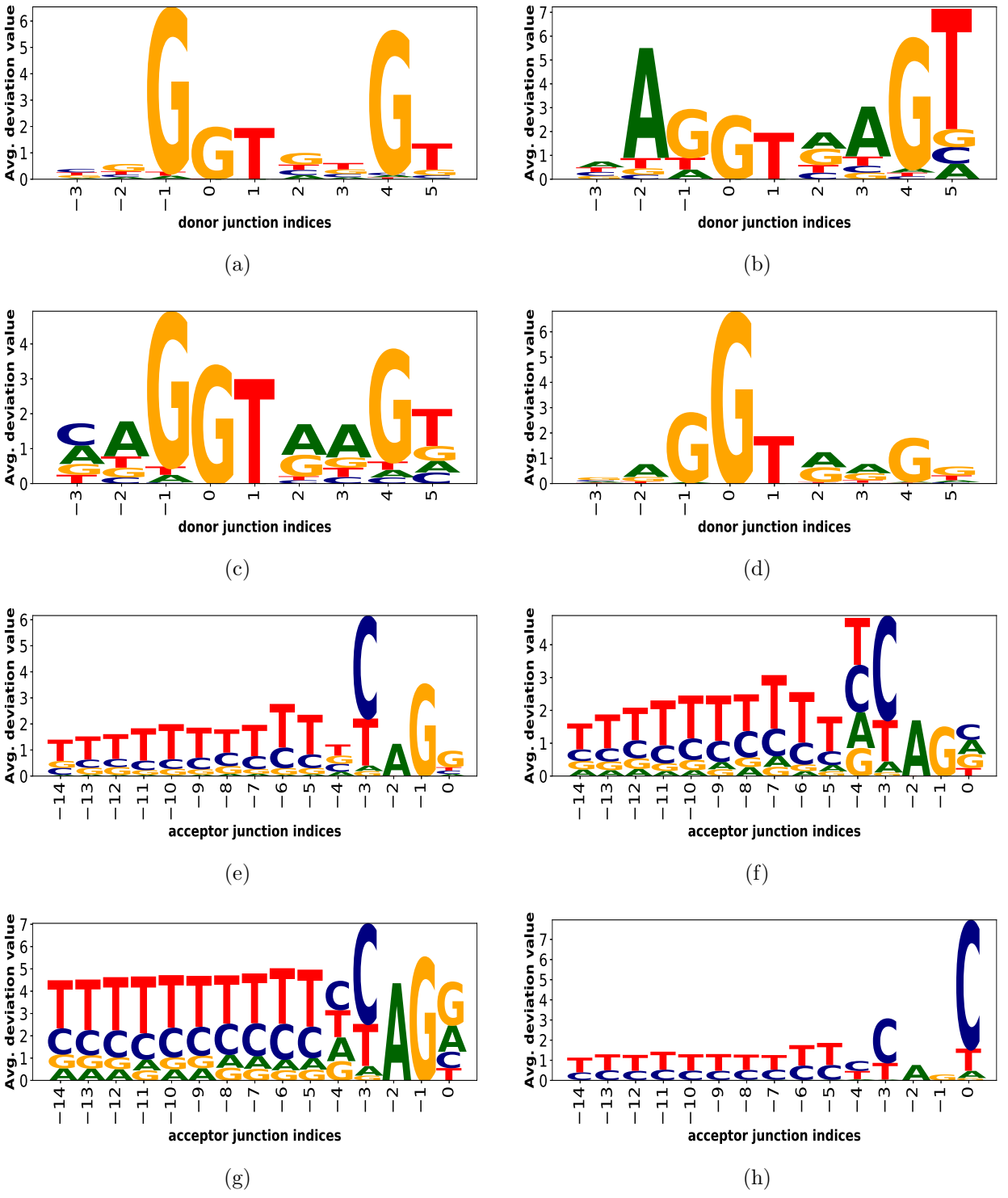
Non-canonical motifs: Two existing features of non-canonical splice junctions are observed in our visualization analysis. Both the back-propagation based visualizations (smooth gradients (Figure 5.10(a)) and integrated gradients (Figure 5.10(b))) pick up GC as an alternative donor site consensus. This consensus has also been observed in a previous study, where the most common class of non-consensus donor splice junction has been reported as GC [93].

Both occlusion-3 (Figure 5.10(c)) and smooth gradients (Figure 5.10(d)) suggest the presence of weaker PY-tract upstream of the non-canonical acceptor junction. This is also in coherence with the observation made in Section 5.4.2.1. Additionally, both these methods suggest a weaker presence of nucleotide C at consensus dimer position -1. This complies with the second class of exception ($AT - AC$) to the consensus dimers reported in [93].

5.4.2.3 Identifying the location of branchpoint consensus 'CTRAY'

We plot the average deviation value of the branchpoint pattern 'CTRAY' [95] in -40 to -15 nt upstream region of the acceptor splice junctions. Although the branchpoint motif is highly degenerate, it is usually observed to be strongly conserved towards the last 50 nt of the introns with a peak at around -23 nt relative to the acceptor splice junction [31]. This is also observed in most of the visualization results (Figure 5.11(a), 5.11(b), 5.11(c), and 5.11(d)), where the peak mostly lies between -25 to -20 nt.

The only exception occurs in the case of occlusion-1 (Figure 5.11(e)), where peaks are observed close to -30 and -40 nt upstream. On the other hand, occlusion-3 (Figure 5.11(f)) displays the peak near -20 nt. This observation complies with the fact that in non-linear models like deep neural networks, the result of occlusion is strongly influenced by the number of features occluded. The occlusion visualization focuses on the key features when larger occlusion windows are considered [7]. Branchpoint peak is observed within a similar range (-25 to



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 5.8: The splicing motifs at canonical splice junctions. The average deviation value per position per nucleotide is shown for canonical donor junction motifs obtained from (a) occlusion-1 (b) occlusion-3 (c) omission (d) integrated gradients and canonical acceptor junction motifs obtained from (e) occlusion-1 (f) occlusion-3 (g) omission and (h) smooth gradients.

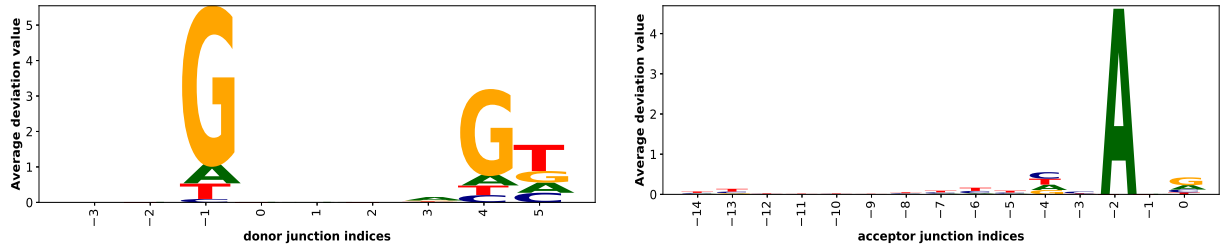


Figure 5.9: Average deviation value per position per nucleotide based on attention weights of canonical sequences.

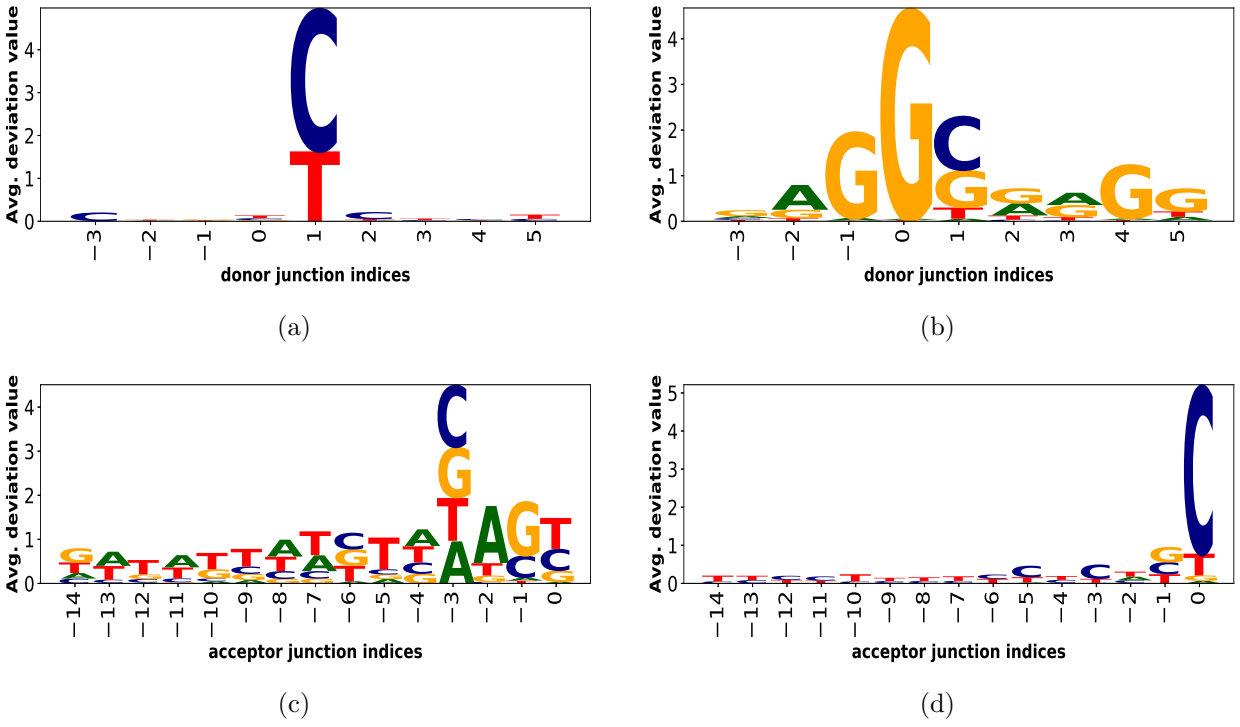


Figure 5.10: The splicing motifs at non-canonical splice junctions. The average deviation value per position per nucleotide is shown for non-canonical donor junction motifs obtained from (a) smooth gradients and (b) integrated gradients as well as non-canonical acceptor junction motifs obtained from (c) occlusion-3 and (d) smooth gradients.

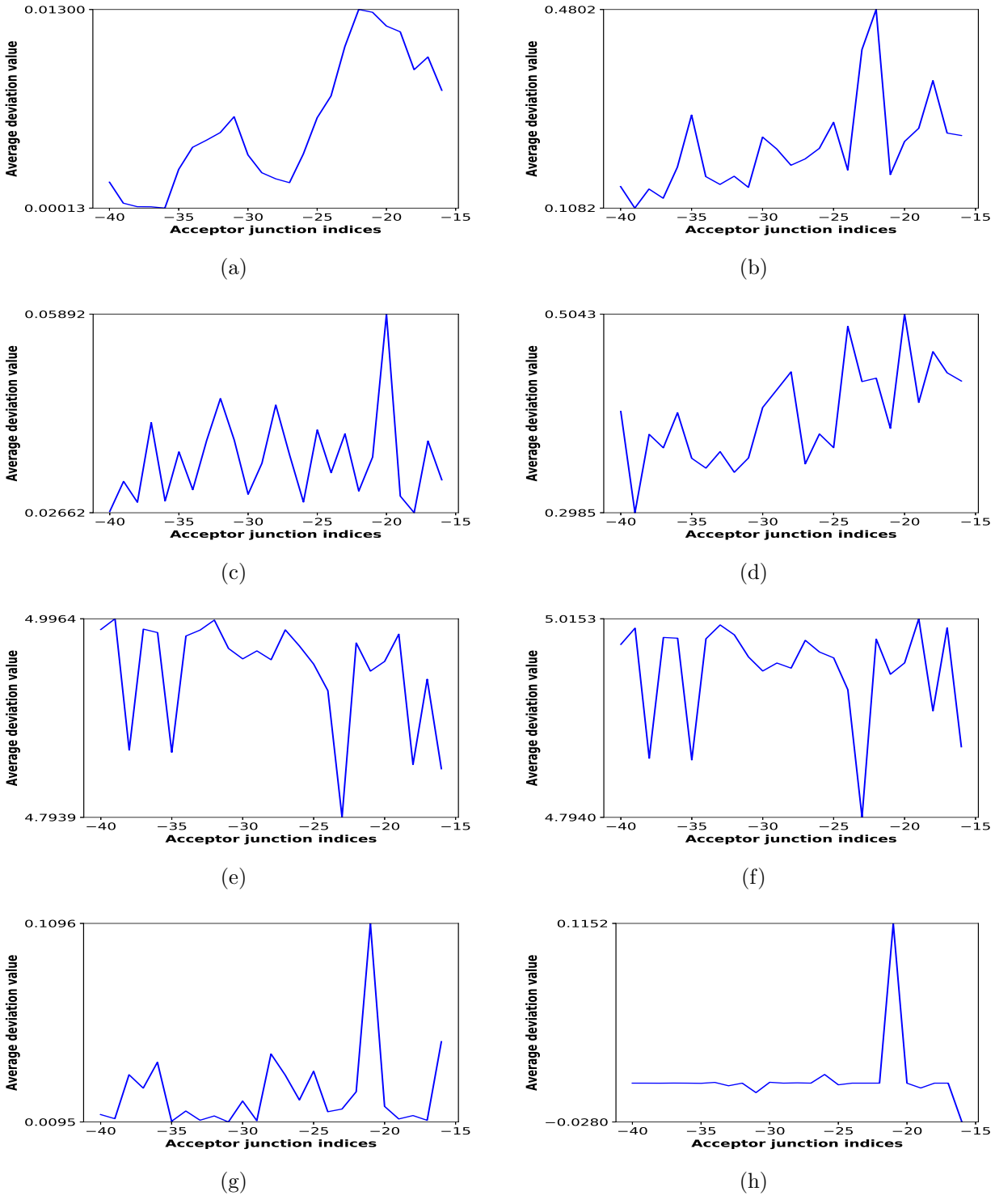
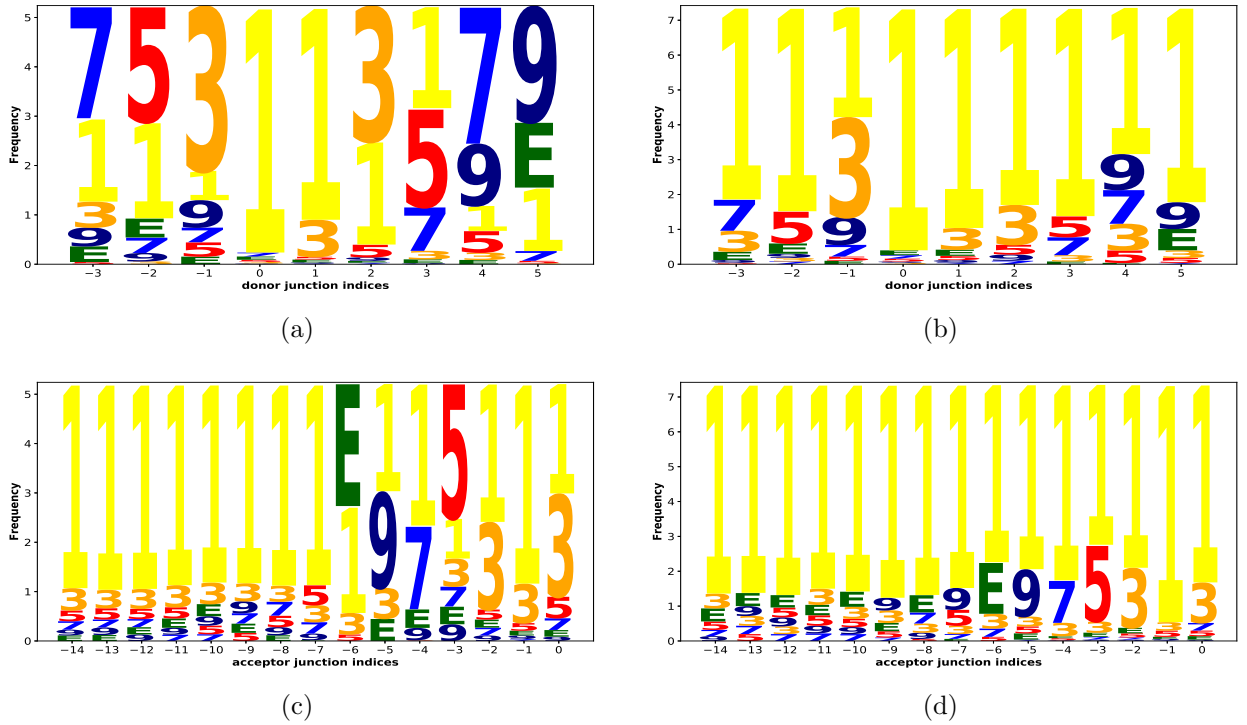


Figure 5.11: The average deviation value of branchpoint pattern CTRAY in the upstream region [-40, -15] of acceptor splice junction. The average deviation value per position for the pattern CTRAY is shown for (a) attention (b) smooth gradients (c) integrated gradients (d) omission (e) occlusion-1 (f) occlusion-3 of canonical acceptor splice junctions (g) integrated gradients and (h) occlusion-3 of non-canonical acceptor splice junctions.



(a) (b) (c) (d)

Figure 5.12: The optimal motif length per position. The frequency of different window lengths, varying from 1 to 11 (represented by E), is shown for occlusion of canonical (a) donor and (c) acceptor splice junctions and occlusion of non-canonical (b) donor and (d) acceptor splice junctions.

-20 nt) in non-canonical acceptor splice junctions as well. Figure 5.11(g) and Figure 5.11(h) show the branchpoint peaks obtained from integrated gradients and occlusion-3, respectively.

5.4.2.4 Identifying the optimal motif length per position

To leverage the capability of the occlusion technique to focus on variable length features, we plot the frequency of different window lengths (varying from 1 to 11) with the highest deviation values. We observe that for canonical donor junction (Figure 5.12(a)), both the dimer indices (0, 1) produce the maximum deviation values for window length 1. This observation suggests that these two positions are maximally functional in the identification of donor splice junction. As we move further in the flanking region, the optimal feature length increases up to 9. This suggests that the flanking nucleotides are weaker signals that require the extended 9-mer motif for recognition of the splice junctions.

The canonical acceptor splice junction (Figure 5.12(c)) shows a similar trend, where optimal feature length goes up to 11 nt (denoted by 'E') in the PY-tract region. However, the PY-tract majorly displays 1 nt as the optimal feature length, suggesting that the PY-tract is another key feature for acceptor junction identification [95].

Table 5.5: Summary of the various visualization techniques in their ability to identify selected canonical splicing features.

Features	Attention	Occlusion	Omission	Smooth gradients	Integrated gradients
Importance of sequence position	✓	✓	✓	✓	✓
Donor site consensus	✗	✓	✓	✗	✓
Acceptor site consensus	✗	✓	✓	✓	✗
Branchpoint	✓	✓	✓	✓	✓

On the contrary, non-canonical donor and acceptor junctions (Figure 5.12(b) and 5.12(d)) consistently focus 1 nt as the optimal feature length across the junction region. This suggests that the non-canonical junction identification does not primarily depend on consecutive nucleotides in the vicinity of splice junctions but possibly on different individual nucleotides in that region. This complies with the observation in Section 5.4.2.1. We can justify this observation with the study [95] which shows that non-canonical splice junctions usually lack one or more known canonical consensus, or the consensus may be distally located. This results in the existence of novel recognition pathways for non-canonical splicing.

5.5 Discussion

We perform a comprehensive analysis of various visualization techniques based on the extraction of several known splicing features. The observations made on the ability of the visualization techniques, in the case of canonical sequences, are summarized in Table 5.5. Although the visualizations display comparable performances in most of the selected features, we still conclude that perturbation based visualizations (occlusion and omission) are the most consistent in their performance across all the known motifs in canonical sequences.

However, in the case of non-canonical sequences, we observe a mixed performance of perturbation and back-propagation based visualizations in capturing the most commonly occurring non-canonical consensus known in the literature. The understanding developed in this work regarding non-canonical splicing is far from complete.

We have the existing knowledge that non-canonical splicing signals may partially lack the known consensus, or the signals may be scattered far from the splice junctions [95]. We also observe earlier in Section 5.4.2.4 that non-canonical splicing signals may be in the form of various nucleotides dispersed across the genomic region rather than consecutive nucleotides. These results suggest the presence of non-canonical splicing signals far beyond the 40 nt flanking region considered in this work. Therefore, it would be interesting to explore a larger context, especially in non-canonical sequences, to develop a better understanding of the splicing phenomenon as a whole. In the next chapter, we will explore non-canonical

splicing phenomenon in further detail.

5.6 Chapter Summary

In this work, we achieve state-of-the-art performance by application of BLSTM network with attention for the prediction of splice junctions. We redesign some of the existing visualization techniques to be capable of comprehending genome sequences as input for inferring biological features relevant to canonical and non-canonical splicing. We validate the splicing motifs inferred from canonical and non-canonical sequences by comparing it with the consensus known from the literature. We also summarize the performances of various visualization techniques on selected splicing features.

"Nature holds the key to our aesthetic, intellectual, cognitive and even spiritual satisfaction."

Edward Osborne Wilson (1929)
American biologist

6

SpliceViNCI: Visualizing the splicing of non-canonical introns through recurrent neural networks

6.1 Introduction

This chapter is targeted towards the third objective of this thesis. The primary contributions of this chapter are to analyze the neural model in the context of extended flanking regions for non-canonical splice junctions and extract the biologically relevant non-canonical sequence features. Towards these objectives, two ways to generate negative data, and their influence on the model's performance are also analyzed. In particular, we focus on the following research questions:

Question 1: How well can a representation learning model encode non-canonical splicing context?

Question 2: What meaningful and biologically relevant features can be extracted from such models?

Question 3: What can be said about the extracted features in contrast to the known knowledge about the non-canonical splice sites?

Towards answering the research questions mentioned above, this chapter introduces SpliceViNCI, a model that identifies splice junctions by applying bidirectional long short-term memory (BLSTM) networks. Similar models have already been applied by some of the previous research works to identify splice junctions. [37, 74]. Our research endeavor differs from the previous applications on the following factors.

The first factor is the extraction of biologically relevant features learnt by the model for both canonical and non-canonical splice junctions. Most of the previous studies focused only on the canonical splice junctions. Byunghan et al.[74] did consider the prediction of non-canonical splice junctions as well but did not extract any biologically relevant features.

However, our study extensively and systematically analyzes the non-canonical splicing phenomenon through visualization of the features learnt by the model. Our previous work, namely SpliceVisuL[37], did focus on identification and feature extraction of non-canonical splice junctions as well but considered a limited flanking region of 40 nt.

Furthermore, Dutta et al.[37] inferred the presence of subtle non-canonical splicing features beyond the 40 nt context. The inference was based on the combination of the results from SpliceVisuL analysis and the existing knowledge. SpliceVisuL analysis indicated that non-canonical splicing signals are in the form of isolated nucleotides rather than consecutive motifs. This was further supported by the existing knowledge that non-canonical splicing signals partially lack the known consensus, and often the signals are scattered far from the splice junctions.[95] SpliceViNCI explores the hypothesis of the presence of signals in an extended region by considering a larger context in the vicinity of splice junctions.

SpliceViNCI targets to attain optimal performance for the identification of canonical as well as non-canonical splice junctions. We extract the non-canonical splicing features learnt by the model and validate them with the existing knowledge. Furthermore, the attention layer present in SpliceVisuL is removed from SpliceViNCI since the attention layer failed to capture all the relevant features in SpliceVisuL.[37] It is also worth noting that, unlike spliceVisuL, SpliceViNCI is trained with the negative dataset that comprises both canonical and non-canonical splice junctions. This is done so that the model can recognize the more subtle non-canonical splicing signals apart from the signals at the junctions.

The contributions of this work can be summarized as follows:

- We present a BLSTM model named SpliceViNCI, which attains state-of-the-art performance for the identification of both canonical and non-canonical splice junctions.
- We design two datasets, Type-1 and Type-2, based on two different sampling strategies to generate negative data.
- We analyze the performance of various state-of-the-art models as well as the proposed model with both the datasets.
- We analyze the length of the flanking region required for obtaining the optimal performance in the identification of non-canonical splice junctions.
- We apply two effective visualization techniques to discern the non-canonical splicing features learnt by the model. The findings thereof are validated with the existing knowledge from the literature.

6.2 Methods

This section elaborates on the neural network architecture employed for the classification of true and decoy splice junctions. We subsequently describe the visualization techniques

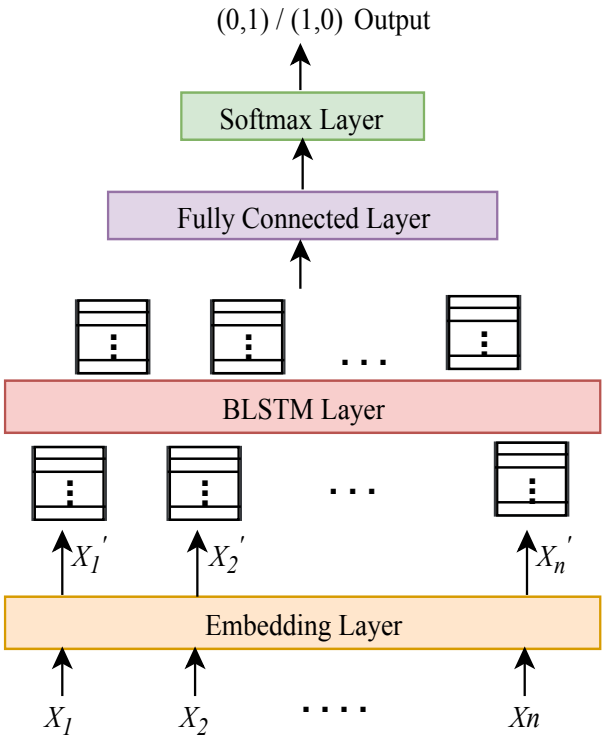


Figure 6.1: A schematic of the network architecture.

applied to extract the relevant features learnt by the model.

6.2.1 Neural architecture

The overview of the network architecture is shown in Figure 6.1.

1. **Input representation:** The input to the learning model is the genome sequences extracted from the vicinity of splice junctions. These sequences comprise the four nucleotides: A (*Adenine*), C (*Cytosine*), G (*Guanine*), T (*Thymine*) and N (*denoting any one of the four nucleotides*). The input sequences are passed through an embedding layer to generate a k-dimensional dense representation for each of the five nucleotide codes. Therefore, an input sequence of length n will be transformed into an $n \times k$ dimensional dense vector that gets updated while training the network. The dense vector is observed to perform better than the traditional one-hot encoded vector due to the learning of meaningful representation through training.[74]

2. **Splice junction representation using BLSTM network:** The BLSTM layer comprises the heart of SpliceViNCI, which captures relevant features from the input sequences. The architecture and working principle of the BLSTM layer is same as proposed in SpliceVisuL (Section 5.2.2.2).

6.2.2 Visualization techniques

We employ two effective visualization techniques, namely integrated gradients and occlusion, for the extraction of relevant splicing features learnt by the model. Both the visualization techniques assign an importance value to each nucleotide position in a genome sequence. We name the importance value as *deviation value*. Higher *deviation value* for a sequence position implies that position as more significant in the identification of the splice junction. The visualization techniques have been explained in Section 5.2.3.

6.3 Experimental Setup

6.3.1 Dataset

The procedure of generating the positive and negative data is described in the following subsection.

6.3.1.1 Positive data

We assess the performance of SpliceViNCI on the identification of novel splice junctions. We, therefore, generate the training and test dataset from two different versions of GENCODE [47] annotations based on human genome version *GRCh38*. Each sample in the data is an intron that comprises donor-acceptor junction pair. The introns are extracted from protein-coding genes only.

290,502 junction pairs are extracted from version 20 as the training data, whereas 293,889 splice junctions are extracted from version 26. The test data is composed of only those introns which were not annotated in version 20. This yields 5,612 novel junction pairs in the test data. We consider introns of length greater than 30 nt only since an existing study [102] suggests that introns of length less than 30 nt can be attributed to sequencing errors. Further, each intronic sequence is truncated to a fixed length by chipping and concatenating the donor and acceptor junctions with a certain length of flanking region (see Section 6.4.2).

6.3.1.2 Negative data

Based on the type of features captured in the data, two variants of negative data is generated. Both randomness-based and consensus-based negative data are described in the following section.

Randomness-based negative data: We extract a subsequence from the center of an intron with the safe assumption that no splice junction will be present between a pair of

donor-acceptor junctions. This procedure of negative data generation is proposed by Noordewier et al..[99] The lengths of positive data and the extracted subsequence are taken equal. The non-randomness in genome sequences is captured in this type of negative data. We obtain $290,502$ false samples for training data and $5,612$ false samples for test data using this procedure.

Consensus-based negative data: Since we are seeking a deeper insight into non-canonical splicing, therefore, it is necessary to mimic similar sequences in the positive and negative data, so the model learns to identify the signals regulating non-canonical splicing in particular. Existing study says that more than 98% of splice junctions are canonical, comprising the GT and AG consensus dimers.[24] The genome sequences have frequent occurrences of the consensus dimers, but not all of those are identified as splice junctions by the splicing mechanism. This indicates the presence of other splicing signals that govern the selection of splice sites. Apart from the canonical consensus dimers, there are two other commonly known classes of non-canonical splice junctions, namely the GC-AG and AT-AC junction pairs.[93]

Hence, we compose this type of negative data considering randomly selected GT-AG, GC-AG, and AT-AC dimer pairs in the genome sequence such that the dimers are not actual splice junctions. The idea of training the splice site prediction models with a consensus-based negative dataset has been applied in previous works. [16, 75, 142] These works have used datasets like $NN269$ [16] and GWH [75, 120, 142] where the sequences in the negative data comprise only GT and AG dimers at the donor and acceptor splice junctions, respectively. We added two commonly known non-canonical consensus pairs GC-AG and AT-AC to let the model learn features governing non-canonical splicing as well.

We randomly search for the donor site consensus in the genome sequence, followed by the corresponding acceptor site consensus. We name such junctions as the negative splice junctions. The randomly sampled dimers should both lie in the same chromosome. The length of the flanking region is considered equal to that in the positive data. We randomly sample $290,502$ training and $5,612$ test data from the human genome assembly version *GRCh38* using this method. The frequencies of both the classes of non-canonical consensus are considered as 0.5% based on their frequencies reported by Mount et al..[93] The remaining negative data comprises the canonical GT-AG dimer pair.

The distribution of canonical and non-canonical splice junctions in the positive and negative data of both training and test dataset is shown in Table 6.1. We form two types of dataset: Type-1 and Type-2. The Type-1 dataset comprises the positive data and randomness-based negative data, whereas the Type-2 dataset comprises the positive data and consensus-based negative data.

The distribution of the two most frequent non-canonical dimer pairs in the training and test dataset is shown in Table 6.2. In the positive training data, we see that the two

Table 6.1: Distribution of canonical and non-canonical splice junctions in the positive and negative data.

Dataset	# of training samples		# of test samples	
	canonical	Non-canonical	canonical	Non-canonical
Positive	255674	5777	5241	371
Randomness-based negative	1096	260355	14	5598
Consensus-based negative	258939	2512	5557	55

most frequent non-canonical dimer pairs conform to the non-canonical dimers reported in [93]. The frequency depicts the count and percentage of non-canonical sequences comprising a particular dimer-pair. As expected, the randomness-based negative data has a smoother frequency distribution of dimer-pairs since it is not governed by any biases and, therefore, only represents the randomness of a genome sequence.

Table 6.2: Distribution of the top 2 most frequent non-canonical dimer pairs in the positive and negative data.

Dataset	Training data		Test data	
	Top 2	Frequency	Top 2	Frequency
Positive	GC-AG	3848 (66.61%)	GC-AG	165 (44.47%)
	AT-AC	232 (4.02%)	GA-AG	16 (4.31%)
Randomness-based negative	TT-TT	2461 (0.94%)	TT-TT	54 (0.96%)
	AA-TT	2013 (0.77%)	TG-TT	47 (0.84%)
Consensus-based negative	GC-AG	1259 (50.12%)	GC-AG	28 (50.90%)
	AT-AC	1253 (49.88%)	AT-AC	27 (49.10%)

6.3.2 Training and hyperparameter tuning

The training data is partitioned into 90% train and 10% validation data for tuning the hyper-parameters of SpliceViNCI. Each nucleotide in the input sequence of length N is converted to a 100-dimensional vector by the embedding layer to form an $N \times 100$ dense vector. This dense vector is passed through the BLSTM, fully connected, and softmax output layer with 100, 1024, and 2 units, respectively. The values for dropout and recurrent dropout are tuned

to 0.5 and 0.2, respectively. We train the model for 10 epochs with a batch size of 128.

6.3.3 Baselines

Our choice of baselines is based on the existing state-of-the-art models in the task of splice site prediction. SpliceRover[142] is a CNN based state-of-the-art model that predicts splice sites and identifies the relevant biological features. The objectives of SpliceRover aligns with the objectives of SpliceViNCI. We have also chosen another CNN based model, namely DeepSplice[139], as a baseline since both DeepSplice and SpliceViNCI formulate the input data in the form of donor-acceptor junction pair. Therefore, it is interesting to compare the performance of both these models for the same task.

Apart from these, we replaced the BLSTM units of SpliceViNCI with LSTM units and considered the LSTM-base model as a baseline. This performance comparison can justify the choice of BLSTM units as the hidden layer of the model. Finally, we choose SpliceVec-MLP[38] as a baseline because of its promising performance over CNN based model. However, SpliceVec-MLP has a limitation of not being able to extract the biological features governing the model performance. We have not considered SpliceVisuL[37] as a baseline since the model in SpliceVisuL is similar to that of SpliceViNCI, except that the attention layer is removed in SpliceViNCI. The attention layer is removed because the layer failed to recognize meaningful features in SpliceVisuL.

The following models are implemented as baselines and the hyperparameters are tuned using the procedure mentioned in Section 6.3.2. The baselines are trained and tested on the dataset explained in Section 6.3.1. The tuned hyperparameters and the baseline architectures are as follows:

1. LSTM-base: We replace the BLSTM units in SpliceViNCI with LSTM units. All hyperparameters are the same as that of SpliceViNCI.

2. SpliceRover: This model is a deep CNN proposed by Zuallaert et al.[142] The model identifies acceptor (donor) splice junctions in an acceptor (donor) classification model. The authors propose the use of a different number of convolutional layers for different sequence lengths. We consider two convolutional layers, followed by a max pooling layer based on the optimal performance obtained on our dataset. Tuned values of batch size, learning rate, decay rate, number of steps, and Nesterov momentum are 64, 0.05, 0.5, 5, and 0.9, respectively.

3. SpliceVec-MLP: This model is proposed by Dutta et al.[38] The model generates distributed representations of true and decoy splice junctions using a shallow neural network, which is then classified by a multilayer perceptron.[38] The batch size and learning rate of Adam optimizer are considered as 128 and 0.001.

4. DeepSplice: Zhang et al. proposed this model.[139] This is a deep CNN model that identifies a true donor-acceptor junction pair from a decoy junction pair sequence. The values for batch size, epochs, and Adam optimizer learning rate are tuned to 160, 30, and 0.001, respectively.

6.4 Results

6.4.1 SpliceViNCI learns better representations of non-canonical splice junctions

The quality of the embeddings obtained from the fully connected layer of SpliceViNCI is evaluated by projecting the 1024 dimensional dense vectors into 2 dimensional space using t-SNE.[84] We plot the positive and negative non-canonical test sequences from Type-1 and Type-2 dataset in Figures 6.2(a) and 6.2(b), respectively. The points in blue are positive non-canonical sequences, whereas the points in red are negative non-canonical sequences. Similarly, we also plot the embeddings obtained from fully connected layer of SpliceRover for both Type-1 and Type-2 dataset in Figures 6.2(c) and 6.2(d).

Relatively more distinct clusters are observed for both the positive and negative datasets in the case of SpliceViNCI compared to SpliceRover. Since no user-defined features are fed into the model and both the models learn the relevant features *de novo* from the genome sequences, we can infer that the proposed architecture extracts the splicing features better than SpliceRover in the case of both positive and negative non-canonical splice junctions.

6.4.2 Non-canonical splicing features are relatively further from the splice junctions

Since the length of the flanking region containing important splicing signals is not known, we vary this length in the input sequences to find the optimal flanking region that produces the best performance in splice junction prediction. We vary the flanking region from 20 to 180 nt with a step size of 20 nt. An input sequence comprises upstream and downstream regions of donor and acceptor junctions concatenated in order. Each junction comprises a canonical or non-canonical dimer. Therefore a flanking region of length N results in an input sequence of length $4 \times N + 4$.

Table 6.3 shows the performance of SpliceViNCI in terms of F1-score with a varying flanking region on Type-1 and Type-2 dataset. We obtain the performance of SpliceViNCI on canonical (*can*) and non-canonical (*non-can*) splice junctions separately. This is because the splicing signals may be differently distributed for canonical and non-canonical splicing resulting in different optimal flanking regions in both the cases.

We see that the performance of SpliceViNCI in the prediction of non-canonical splice junctions improves by about 14% for Type-1 dataset and 4% for Type-2 dataset with an in-

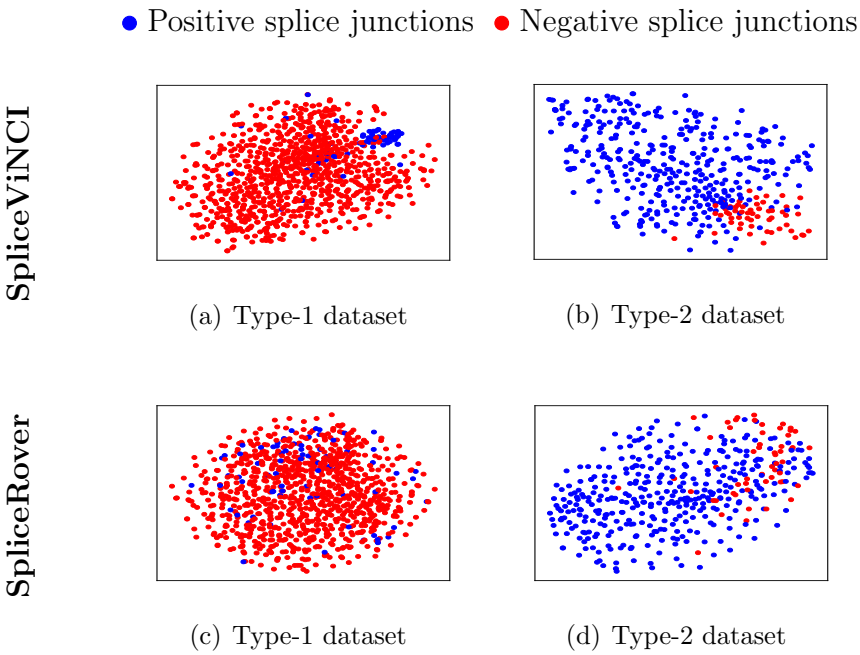


Figure 6.2: t-SNE plots of non-canonical splice junctions obtained from SpliceViNCI for (a) Type-1 dataset, (b) Type-2 dataset and from SpliceRover for (c) Type-1 dataset, (d) Type-2 dataset. The points in blue are positive splice junctions whereas points in red are negative splice junctions.

crease in the context. We also observe that SpliceViNCI obtains comparable performance for both the dataset in case of canonical splice junctions. However, in the case of non-canonical splice junctions, the performance improves significantly in the Type-2 dataset.

SpliceViNCI obtains the maximum F1-score of 97.67% (74.04%) in the Type-2 (Type-1) dataset for the prediction of non-canonical splice junctions. The improvement in performance for the Type-2 dataset can be attributed to the consensus-based negative data in the Type-2 dataset. Since the consensus-based negative data is composed of the splice junction consensuses GT-AG, GC-AG, and AT-AC, the positive and negative data in the Type-2 dataset look very similar at the splice junction. This enables the model to recognize other subtle features in the flanking region apart from the dimers at the splice junctions, which differentiate the true and decoy splice sites. This hypothesis corroborates the rationale behind the similar formulation of negative datasets by Bretschneider et al.[21]

On the contrary, the negative data in the Type-1 dataset comprises sequences from the center of each intron. This type of negative data lacks the consensus dimers and only captures the non-randomness of genome sequences. Therefore, the model recognizes the consensus dimers as the primary features in this type of dataset and may miss out on the other subtle splicing signals that govern non-canonical splicing.

We obtain the optimal performance for the prediction of canonical splice junctions at

Table 6.3: F1-score (in percentage) obtained by SpliceViNCI in identification of canonical (*can*) and non-canonical (*non-can*) splice junctions with varying flanking region on Type-1 and Type-2 dataset.

Flanking region	Type-1 dataset		Type-2 dataset	
	can	non-can	can	non-can
180	99.50	69.71	99.13	97.38
160	99.39	65.47	99.06	97.24
140	99.65	72.09	99.05	97.66
120	99.65	**74.04**	99.05	**97.67**
100	99.67	73.56	99.04	96.82
80	**99.70**	70.31	99.02	96.81
60	99.65	71.06	**99.07**	96.40
40	99.60	69.32	98.30	95.65
20	98.60	60.01	95.82	93.65

60 to 80 nt context. An optimal length of 30 to 40 nt is suggested in the literature [101, 106] and validated in various studies [37, 38, 75, 138, 139] for canonical splice junction prediction. These studies obtained negligible improvement in the model's performance on further increase of the flanking context.

We obtain the optimal performance for the prediction of non-canonical splice junctions at a flanking region of 120 nt. To examine the statistical significance of the context length, we performed the student's t-test. We formed two groups, each comprising F1-scores obtained from five different executions, with a flanking region of 120 nt and 80 nt, respectively. The P-values obtained for Type-1 and Type-2 dataset are 0.002 and 0.003, respectively. We consider a P-value < 0.05 as statistically significant.

The statistical significance of variation in the flanking region indicates the presence of non-canonical splicing signals further away from the splice junction. This inference can be validated by the study, which suggests that non-canonical splice junctions may lack some known consensus, or the splicing signals may be distally located from the splice junctions.[95]

6.4.3 SpliceViNCI outperforms state-of-the-art splice junction prediction models

We compute the prediction performance of various state-of-the-art models on Type-1 and Type-2 dataset, considering the optimal flanking region of 120 nt obtained in Section 6.4.2. F1-score is considered as the performance metric. We see that SpliceViNCI outperforms all state-of-the-art models in the prediction of non-canonical splice junctions in both Type-1 and Type-2 dataset (Figure 6.3).

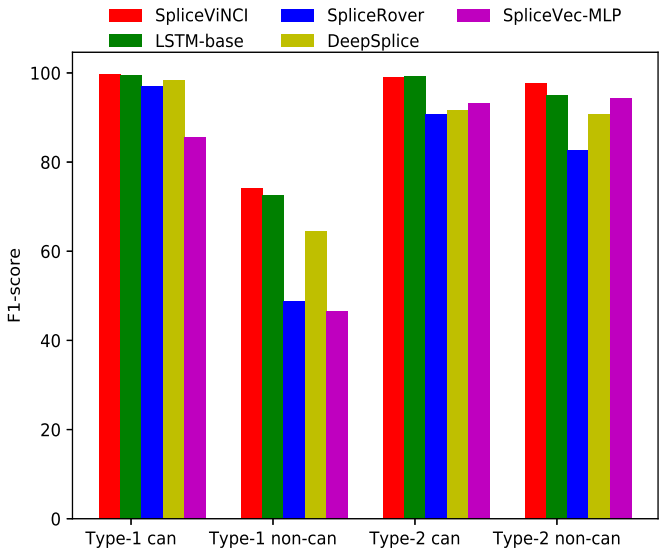


Figure 6.3: Performance of various state-of-the-art models. The performance is measured in terms of F1-score for canonical (*can*) and non-canonical (*non-can*) splice junctions from both Type-1 and Type-2 dataset. F1-score is computed in percentage.

In the prediction of canonical splice junctions, LSTM-base performs comparably to SpliceViNCI for both the dataset. In the prediction of non-canonical splice junctions from the Type-1 dataset, SpliceViNCI shows a minimum improvement of 2% over LSTM-base and a maximum improvement of 28% over SpliceVec-MLP. In the case of the Type-2 dataset, SpliceViNCI shows a maximum and minimum improvement of 2% and 15% over LSTM-base and SpliceRover, respectively. Furthermore, all the models obtain better performance in the identification of non-canonical splice junctions from the Type-2 dataset compared to the Type-1 dataset. This suggests that the negative data in the Type-2 dataset enables the models to learn better representations of the splicing features, as explained in Section 6.4.2.

6.4.4 Donor and acceptor splicing signals identify the splice junctions cooperatively

We consider donor-acceptor junction pairs as the input sequences instead of only the donor or acceptor junctions. This results in the performance improvement of the predictive model, as shown in Table 6.4 for the Type-2 dataset. We see that all the models except DeepSplice perform better on the prediction of donor junctions compared to acceptor junctions. However, the performance improves significantly when donor-acceptor junction pairs are considered as input suggesting the cooperative mechanism of donor and acceptor splicing signals in splice junction recognition. This is also inferred in various studies which state that the donor and acceptor junctions are not recognized through individual splicing signals but through junc-

Table 6.4: Performance of various state-of-the-art models on Type-2 dataset considering donor, acceptor and donor-acceptor junction pair as input. F1-score (in percentage) is computed as the performance metric.

Model	Junction pair		Donor		Acceptor	
	can	non-can	can	non-can	can	non-can
SpliceViNCI	99.05	97.67	93.90	91.19	91.29	77.91
LSTM-base	99.16	95.00	94.01	89.32	90.18	74.68
DeepSplice	91.66	90.81	90.79	86.89	84.27	88.97
SpliceRover	90.68	82.63	86.37	83.30	84.66	71.24
SpliceVec-MLP	93.23	94.23	79.97	67.98	73.54	59.24

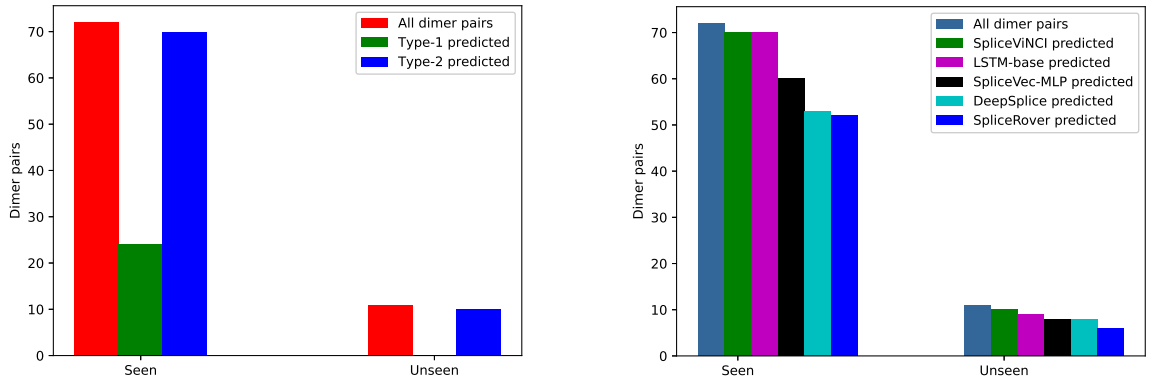
tion pairs across exons or introns.[18, 59, 115]

6.4.5 Identification of novel non-canonical splice junctions by SpliceViNCI

We intend to assess the performance of SpliceViNCI on the identification of novel non-canonical splice junctions, as described in Section 6.3.1. With this objective, we identify two sets of dimer pairs: seen and unseen. The set of *seen* dimer pairs comprise those positive non-canonical splice junction pairs that are present in both training and test data. Whereas, the set of *unseen* dimer pairs comprise those positive non-canonical splice junctions that are present in test data but not in training data.

Figure 6.4(a) shows the total number of dimer pairs present in both *seen* and *unseen* categories. The figure also shows the number of dimer pairs correctly identified by SpliceViNCI as splice junctions from *seen* and *unseen* sets in case of both Type-1 and Type-2 dataset. Since the positive data is the same in both Type-1 and Type-2 dataset, hence the number of dimer pairs belonging to *seen* and *unseen* sets are the same for both the dataset. We observe that SpliceViNCI performs better in the identification of *seen* data in the case of the Type-2 dataset compared to the Type-1 dataset. It is also noteworthy that in case of *unseen* data, SpliceViNCI does not identify any dimer pair from Type-1 dataset, whereas it identifies all except one dimer pairs in case of Type-2 dataset.

We were also curious to observe the performance of all the baselines in the identification of *unseen* data. Figure 6.4(b) shows the number of dimer pairs identified by SpliceViNCI and all the baselines from both *seen* and *unseen* sets in case of Type-2 dataset. We consider the Type-2 dataset for the analysis because the Type-2 dataset can capture more relevant non-canonical features, as shown in Figure 6.3 and explained in Section 6.4.2. It is also fair



(a) (b)

Figure 6.4: Performance of SpliceViNCI on *seen* and *unseen* data. (a) The number of *seen* and *unseen* dimer pairs identified by SpliceViNCI from Type-1 and Type-2 dataset. (b) The number of *seen* and *unseen* dimer pairs identified by SpliceViNCI and all the baselines from Type-2 dataset.

to compare all the baselines with the Type-2 dataset because we see in Section 6.4.3 that all the baselines obtain a better performance in the case of the Type-2 dataset. We observe that SpliceViNCI outperforms all the baselines in the identification of both seen and unseen dimer pairs.

6.4.6 Visualization of splicing features captured by SpliceViNCI

The presence of splicing features in the vicinity of splice junctions facilitate the identification of splice junctions by the prediction model. Interpretation of the features captured by the prediction model is necessary to justify the superior performance of the model. The splicing features analyzed using the visualization techniques are summarized in the subsequent subsections. The visualizations are carried out on the non-canonical sequences from the Type-2 dataset. Since our visualizations are specifically for the non-canonical splice junctions, we have considered the Type-2 dataset, which captures more relevant features for the non-canonical splice junctions.

6.4.6.1 Significance of sequence positions

We plot the average *deviation values* obtained for donor and acceptor junction pairs from integrated gradients (Figures 6.5(a) and 6.5(c), respectively) and occlusion-1 (Figures 6.5(b) and 6.5(d), respectively). A flanking upstream and downstream region of 120 nt is also considered. We observe that both the visualization techniques identify the acceptor and donor splice junctions as the most significant. The importance decreases as the distance

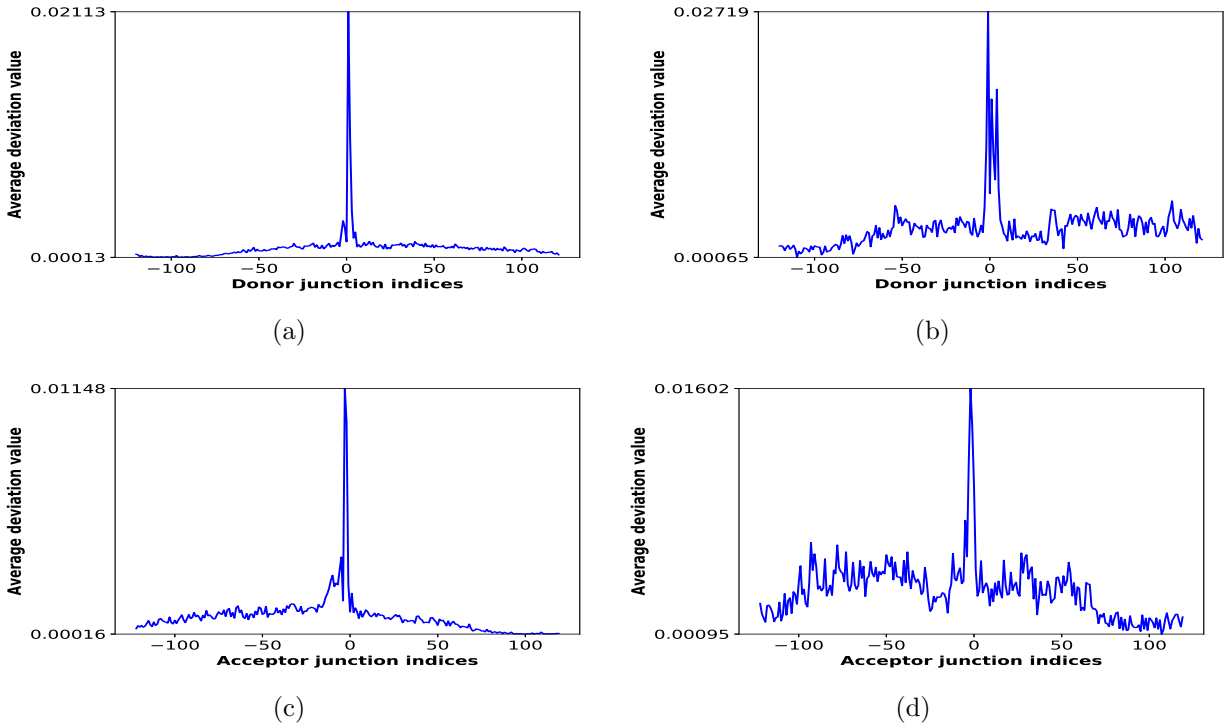


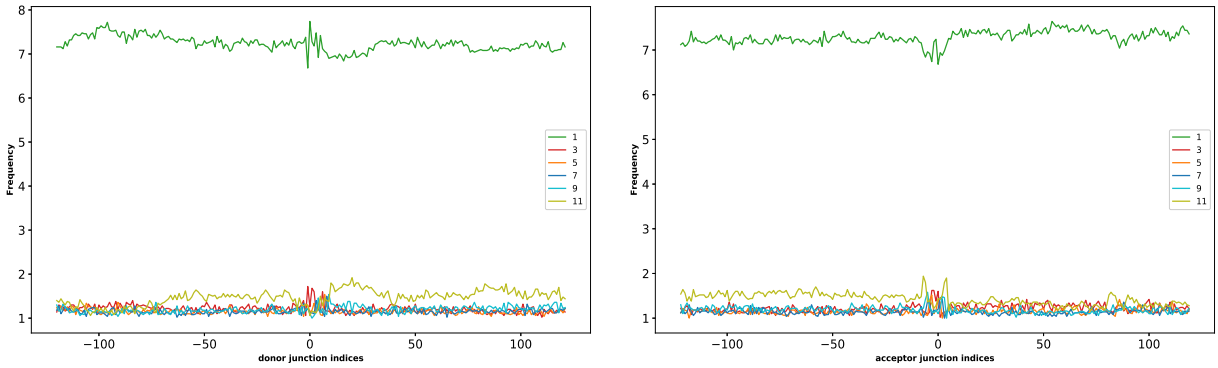
Figure 6.5: The significance of sequence position. The average deviation value per position is shown for non-canonical donor junctions by (a) integrated gradients (b) occlusion-1, and non-canonical acceptor junctions by (c) integrated gradients (d) occlusion-1.

of the nucleotide position increases from the splice junction. It is also noteworthy that the importance of the intronic region is higher than the exonic region at both donor and acceptor junctions.

Additionally, integrated gradients show higher deviation values just upstream of the acceptor junction in the region 0 nt to -15 nt compared to the downstream region, which is due to the presence of weakly conserved polypyrimidine tract (PY-tract) in non-canonical splice junctions.[95] On the other hand, occlusion-1 shows higher importance for sequence positions deeper into the intronic region at the acceptor junction than integrated gradients. This suggests the presence of splicing features upstream of the PY tract.

The higher importance upstream of the PY-tract is captured by occlusion-1 but not integrated gradients. This can be explained by a study [7], which states that occlusion is capable of capturing individual features in isolation, whereas integrated gradients perform better when multiple features are considered together. We reported a similar observation in the previous chapter where isolated nucleotides were captured as splicing features by occlusion, and consecutive nucleotides were captured as features by integrated gradients. If we consider each sequence position as a feature, then higher *deviation value* assigned by occlusion-1 upstream of the PY-tract suggests the presence of dispersed nucleotides in this

region that possibly act as splicing features. This is also stated in [95].



(a) (b)

Figure 6.6: The optimal motif length per position. The frequency of different window lengths, varying from 1 to 11, is shown for occlusion of non-canonical (a) donor and (b) acceptor splice junctions.

6.4.6.2 Optimal feature length per position

As the significance of sequence positions observed from Section 6.4.6.1 suggests the presence of dispersed and isolated splicing features along the intronic region at acceptor junction, we intend to assess the optimal feature length at each sequence position. To this end, we plot the relative frequency of each window length across all sequences when it produced the highest deviation value at a particular sequence position. We observe that at both donor (Figure 6.6(a)) and acceptor (Figure 6.6(b)) splice junctions, the plot consistently displays highest frequency for the window length of 1 nt along the entire sequence. This again validates that the features governing non-canonical splicing are mostly dispersed along the sequence and are not placed at consecutive nucleotide positions.

6.4.6.3 Importance of each nucleotide per position

To access the importance of each nucleotide at each sequence position, we plot the average deviation value per position per nucleotide. Integrated gradients show higher deviation value for C and T along the entire intronic region at both donor (Figure 6.7(a)) and acceptor junctions (Figure 6.7(c)). The deviation values for all the nucleotides diminish along the exonic region.

Occlusion extracts G as the most important nucleotide along the intronic region for both donor and acceptor junctions. The deviation value for G is particularly high in the region from -30 nt to -100 nt upstream of the acceptor junction, as shown in Figure 6.7(d).

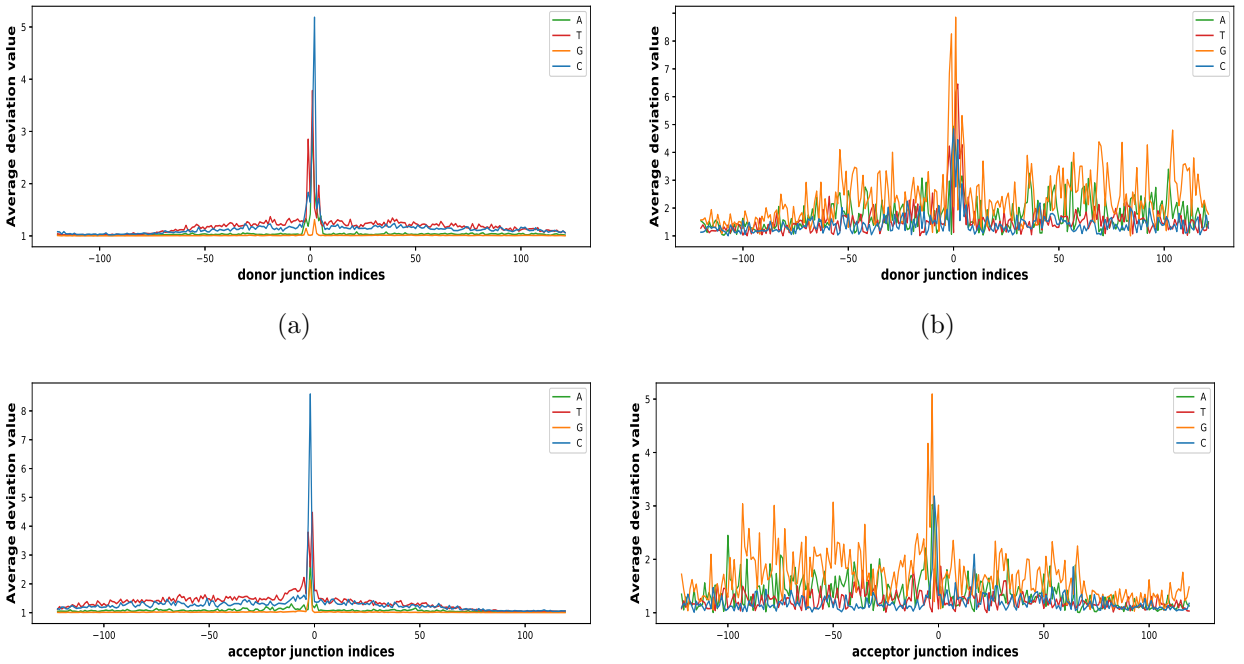


Figure 6.7: The average deviation value per position per nucleotide. The average deviation value per position per nucleotide is shown for non-canonical donor junctions by (a) integrated gradients (b) occlusion-1 and non-canonical acceptor junctions by (c) integrated gradients (d) occlusion-1.

This suggests the presence of a G-rich splicing feature in this region.

The above two observations corroborate the study in [95], which suggests the presence of G-rich motifs upstream of PY-tract in the case of non-canonical introns having weak PY-tract. The integrated gradient captures the weak PY-tract, whereas occlusion captures the G-rich motifs. However, the higher deviation values for C and T at the donor junction does not relate to any relevant knowledge from the literature.

6.4.6.4 Most important motifs in a specific region

Murray et al. explored the region -30 nt to -80 nt upstream of the acceptor junction to identify the enriched motifs that regulate splicing in the case of non-canonical splicing. They characterized the relative enrichment of all 4-7 nucleotide k-mers in the specified region and obtained several G-rich motifs.[95] A similar region is also highlighted in Figure 6.7(d) and described in Section 6.4.6.3.

We conducted a similar analysis by computing the relative frequency of the most important 4-7 nucleotide k-mers in the region -30 nt to -80 nt relative to the acceptor junction.

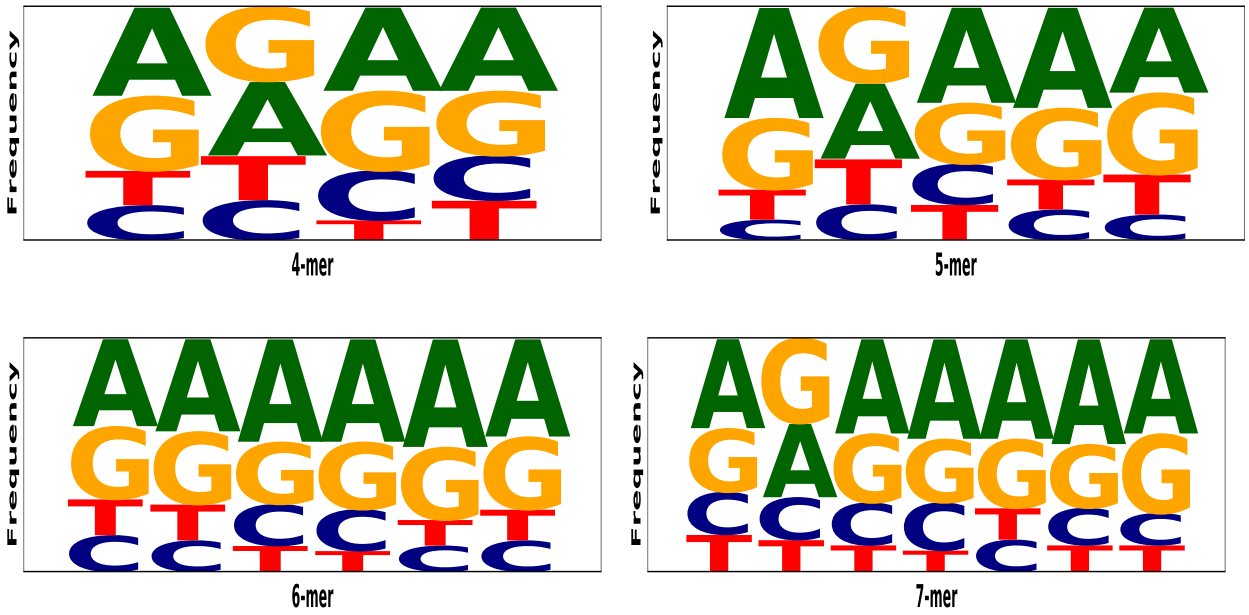


Figure 6.8: The frequency of various k-mers, given by occlusion, in the region -30 nt to -80 nt upstream of the non-canonical acceptor junction.

The most important k-mer in a particular position is the k-mer, which obtains the highest average deviation value, at that position, across all sequences. The motifs obtained for the four different lengths are shown in Figure 6.8. We observe that the specific region is rich in [AG] nucleotides suggesting the importance of purines in the regulation of non-canonical splicing.

To sum up, we can conclude that both the visualization techniques play a vital role in the extraction of non-canonical splicing features. Since the non-canonical splicing features comprise both single and contiguous nucleotides, we can apply both the visualization techniques for the comprehensive understanding of non-canonical splicing.

6.5 Chapter Summary

We apply a BLSTM based prediction model named SpliceViNCI that achieves state-of-the-art performance in the identification of canonical and non-canonical splice junctions. SpliceViNCI outperforms state-of-the-art models in the identification of both annotated and novel splice junctions. Our study finds that a flanking region of 120 nt produces optimal performance in the prediction of non-canonical splice junctions.

We employ two effective visualization techniques for the extraction of relevant splicing features learnt by the model. The visualization techniques are redesigned to be capable of

comprehending genome sequences as input. We employ both back-propagation based and perturbation based visualization techniques to leverage the benefit of both the techniques. The features obtained are validated with the existing knowledge from the literature.

"Nature holds the key to our aesthetic, intellectual, cognitive and even spiritual satisfaction."

Edward Osborne Wilson (1929)
American biologist

7

SpliceTrans: Transferring knowledge across species for identification of splice junctions

7.1 Introduction

This chapter is based on the fourth objective of the thesis. In the previous chapters, we focused on identifying splice sites from a single species. This chapter focuses on identifying splice sites from multiple species. In the recent times, several deep learning models [4, 37, 38, 59, 74, 75, 127, 138, 139, 142] have been applied to the task of splice site identification such that the model can learn and capture the relevant features from the genomic sequences by itself. However, most of the studies are based on studying and identifying splice sites in a single species. Some of the models study multiple species like Homo sapiens, Arabidopsis thaliana, Oryza sativa japonica, Drosophila melanogaster, and C. elegans [4, 127, 142].

However, Zuallaert et al. [142] do not discuss the generalization capability of their proposed model by training on one species and testing on another. Albaradeia et al. [4] proposed a model named Splice2Deep which trains and tests on different species and still identifies the splice sites with high accuracy. However, they do not extract or discuss any biological features learnt by the model. Furthermore, they consider mononucleotide and trinucleotide data representation for intronic and exonic features, respectively, considering intronic region to be non-coding. However, intronic regions are sometimes retained in the mature mRNA to be subsequently translated to proteins. Wang et al. [127] proposed SpliceFinder, which trains a CNN model to identify acceptor, donor, and false splice sites. Although SpliceFinder trains and tests the model on both canonical and non-canonical splice sites of various species, it extracts and visualizes features from canonical sites only. Furthermore, the performance of SpliceFinder is tested on the identification of splice sites extracted from the same version as that of the training dataset. Hence, SpliceFinder is not tested on the task of identifying novel splice sites.

To alleviate the limitations of the current research works discussed above, we apply neural network models to identify novel canonical and non-canonical splice sites from various

species. In particular, we ask the following questions in this work:

Question 1: Whether various neural models perform equally well on identifying splice sites from multiple species? Does any particular model outperform the rest?

Question 2: Can the neural models be used to annotate a poorly studied species using data from an extensively annotated species?

Question 3: What are the canonical and non-canonical splicing features extracted from the different species?

To answer the above questions, we apply a BLSTM model to identify splice sites in various species. We name the model as *SpliceTrans*. The performance of SpliceTrans is assessed in both the conditions when the training and the testing dataset may be from the same species or different species. We compare the performance of SpliceTrans with state-of-the-art models to evaluate whether any particular model performs better than the rest in this task. We also augment the dataset by combining training data from more than one species to observe the improvement of the model's performance in the task of canonical and non-canonical splice site identification. The improvement in a model's performance on augmenting training data from another species suggests that the model can identify splice sites from poorly annotated species using data from another extensively annotated species.

Moreover, SpliceTrans is tested on novel splice sites such that the training and testing samples are extracted from two different versions of the dataset. The ability to identify novel splice sites indicates that a model is generalizable. In the previous two chapters, a BLSTM model has already been applied to identify and visualize splice sites within a single species [37, 39]. However, the generalizability of a BLSTM model has not been exploited so far in identifying splice junctions from multiple species. This chapter works towards this objective. Finally, we extract the canonical and non-canonical features captured by SpliceTrans from the various species and validate them with the existing knowledge.

The contributions of this chapter can be summarized as:

- SpliceTrans outperforms the state-of-the-art models in identifying splice sites from the human, mouse, and drosophila melanogaster species.
- SpliceTrans also identifies canonical and non-canonical splice junctions from species on which the model is not trained.
- SpliceTrans maintains its superior performance on imbalanced data. This makes it more robust and applicable in the annotation of multiple species.
- We observe that augmenting the training dataset of one species with that of another species improves the performance of the model, especially in identifying non-canonical splice sites.
- We further extract and compare the biological features learnt by the models with different training datasets and validate them with the existing literature.

7.2 Methods

This section discusses the network architecture of the neural model employed in splice site identification from multiple species. The visualization technique used for extraction of the biologically relevant features is also discussed.

1. **Neural architecture:** SpliceTrans is a BLSTM-based neural network model. The input representation and neural architecture of SpliceTrans are the same as that of SpliceViNCI as described in Section 6.2.1 and Figure 6.1.

2. **Feature interpretation:** We apply Integrated gradients [122] for extracting and interpreting the non-canonical splicing features learnt by SpliceTrans. Integrated gradients is a back-propagation based visualization technique proposed by Sundararajan et al.. The technique is discussed elaborately in Section 5.2.3. We choose integrated gradients as the visualization technique because we observe in Section 5.4.2.2 that back-propagation based visualization techniques perform better than perturbation based techniques in the case of non-canonical splice sites.

7.3 Experimental Setup

7.3.1 Dataset

The training and testing dataset are generated for three species, namely *Homo sapiens (Human)*, *Mus musculus (Mouse)*, and *Drosophila melanogaster (Drosophila)*.We choose mouse and drosophila species in order to have a wide range of comparison because mouse is closer to human in the phylogenetic tree, whereas drosophila is further [54]. Mouse and human species have 99% homologous protein-coding genes [141]. In contrast, drosophila and human have 60% homologous protein-coding genes [90]. The procedures of generating the positive and negative data are described in the next section.

7.3.1.1 Positive data

The genome sequence data (FASTA files) and the corresponding annotations (GTF files) are downloaded from the GENCODE database [47] for human and mouse species. The FASTA and GTF files for Drosophila melanogaster are downloaded from the Ensembl database [57].

We test the various models on the identification of novel splice sites. Therefore, the training and testing samples are extracted from two different versions of the database. The training data is extracted from an earlier released version. The testing data is generated from a later release such that the testing samples are not present in the training version of the database. This ensures that the model can identify splice sites that were not annotated in the training version of the dataset, thus making the model more robust.

The number of introns present in the training and testing data of the different species is mentioned in Table 7.1 along with their corresponding reference genome and release versions. We observe that the number of introns in drosophila is much less than in human and mouse. This is because the drosophila genome comprises only four pairs of chromosomes compared to 23 and 20 pairs in human and mouse species, respectively.

Table 7.1: Distribution of positive dataset in mouse, human and Drosophila melanogaster.

Species (reference genome)	Training data # of introns (version)	Testing data # of introns (version)
Mus musculus (GRCm38)	225616 (V:M2)	26030 (V:M24)
Homo sapiens (GRCh38)	290502 (V:20)	5612 (V:26)
Drosophila melanogaster (BDGP6)	58522 (V:95)	118 (V:103)

7.3.1.2 Negative data

The negative dataset is generated by randomly extracting genome sequences from the genome sequence data. We randomly search for the donor site dimer and a subsequent acceptor site dimer. Both the donor and acceptor site dimers are present in the same chromosome and are not annotated as splice sites in the positive dataset. The method of generating the negative data is the same as consensus-based negative data described in Section 6.3.1.2.

The donor-acceptor dimer pairs considered in the generation of negative dataset are *GT-AG*, *GC-AG*, and *AT-AC*. The $GT-AG$ consensus rule applies to all the three species considered here [2, 105]. $GC-AG$ and $AT-AC$ are the most frequently occurring non-canonical dimer pairs in mouse species of our positive data. This corroborates the most frequently occurring non-canonical splice junctions in the human species known from literature [93]. The most frequent non-canonical dimer pair in our drosophila positive data is $GC-AG$ and $GT-TG$, closely followed by $AT-AC$. However, for the sake of uniformity of comparison, we use $GC-AG$ and $AT-AC$ as non-canonical negative dimer pairs across all species.

7.3.2 Training and hyperparameter tuning

We partition the training dataset into 90% train and 10% validation data. Each input sequence comprises 40 nt upstream and downstream regions at both donor and acceptor sites along with the junction dimers. The donor and acceptor site sequences are concatenated to form an input sequence of length 164 nt.

The 164 nt input is fed into SpliceTrans, where the embedding layer converts each nucleotide into a 4-dimensional dense vector. The subsequent BLSTM, fully connected and softmax layer comprises 100, 1024, and 2 units. Other hyperparameters like epochs, batch

size, dropout, and recurrent dropout, are set to 10, 128, 0.5, and 0.2, respectively, after tuning.

7.3.3 Baseline

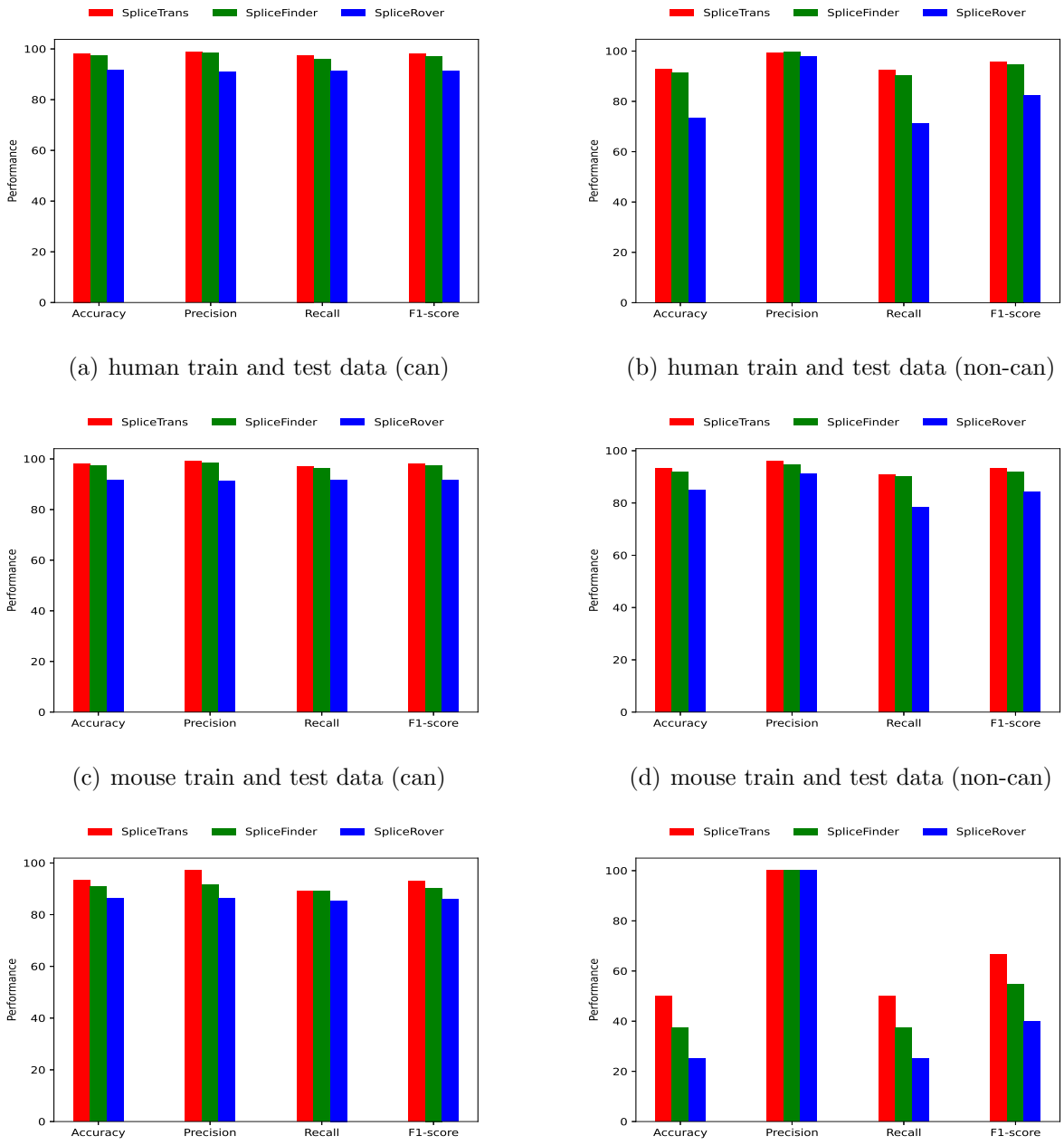
We choose state-of-the-art models as our baselines which have similar objectives as that of SpliceTrans in the task of splice site identification. The following section describes the baseline models along with the hyperparameters used for training and testing the models. The hyperparameters for the baselines are tuned using the procedure mentioned in Section 7.3.2.

1. **SpliceRover:** This is a CNN model comprising several convolutional layers followed by max-pooling, fully connected, and softmax layers. This model is proposed by Zuallaert et al.[142]. SpliceRover identifies the donor and acceptor splice sites in the human and Arabidopsis thaliana species. Additionally, Zuallaert et al. extract the biologically relevant features learnt by the model using the DeepLIFT [117] visualization technique.

   Zuallaert et al. trained SpliceRover with different numbers of convolutional layers for various datasets of variable sequence length. We trained the model with two convolutional layers followed by a max-pooling, fully connected layer, and softmax layer based on the optimal performance on our dataset. Stochastic gradient descent is chosen as the optimizer with learning rate, decay rate, Nesterov momentum, and the number of steps per learning rate decay set to 0.05, 0.5, 0.9, and 5, respectively. The model is trained and tested on the dataset described in Section 7.3.1. Other hyperparameters like epochs and batch size are set to 30 and 64, respectively.

2. **SpliceFinder:** This is a CNN-based model proposed by Wang et al. [127]. The model is trained with the human dataset and identifies donor and acceptor splice sites in several species, namely Drosophila melanogaster, Mus musculus, Rattus, and Danio rerio, without retraining. They also extract the relevant splicing features using the DeepLIFT visualization technique.

   The model comprises one convolutional layer with 50 kernels of length nine followed by a fully connected layer of size 100. A dropout layer is subsequently added with a dropout rate of 0.3. Finally, there is a softmax layer of 2 nodes to classify true and false splice sites. Adam is used as an optimizer with a learning rate of 10^{-4}. Other hyperparameters like epochs and batch size are both set to 50.



(a) human train and test data (can)

(b) human train and test data (non-can)

(c) mouse train and test data (can)

(d) mouse train and test data (non-can)

(e) drosophila train and test data (can)

(f) drosophila train and test data (non-can)

Figure 7.1: Performance (in percentage) obtained by SpliceTrans, SpliceRover and SpliceFinder in the identification of canonical (can) and non-canonical (non-can) splice junctions when trained and tested with data from the same species.

7.4 Results

The results obtained from various analysis of SpliceTrans and their comparison with baselines are discussed in the following subsections.

7.4.1 SpliceTrans outperforms state-of-the-art in identifying splice sites of multiple species

We intend to evaluate the performance of various neural network models on the task of identifying canonical and non-canonical splice sites from multiple species. We use accuracy, precision, recall, and F1-scores as the evaluation metrics. We train and test SpliceTrans, SpliceRover and SpliceFinder prediction models with data from human, mouse and drosophila melanogaster.

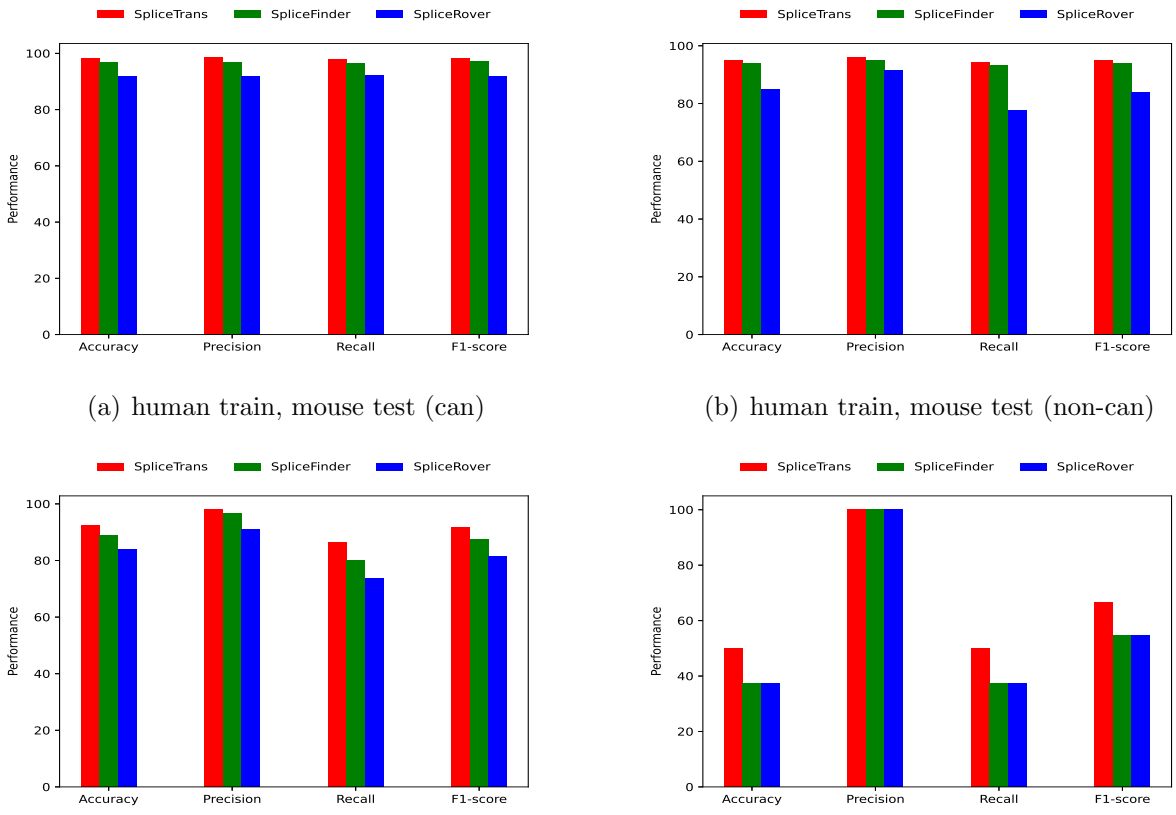
Figure 7.1 displays the performances of SpliceTrans, SpliceFinder, and SpliceRover in the identification of canonical and non-canonical splice sites where each model is trained and tested with the data from the same species. We observe that SpliceTrans outperforms SpliceFinder and SpliceRover in most of the metrics for all three species. This analysis is done to test the consistency of the models in the identification of splice sites in various species.

We observe in Figure 7.1 that SpliceTrans obtains an F1-score approximately 1% to 2.5% more than SpliceFinder in the identification of canonical splice junctions across the three different species. Furthermore, The F1-score of SpliceTrans is 1% to 12% more than SpliceFinder in identifying non-canonical splice sites. The F1-score of SpliceTrans is 7% (13% to 26%) more than SpliceRover in the identification of canonical (non-canonical) splice junctions. Similarly, SpliceTrans outperforms SpliceFinder and SpliceRover in the other three performance metrics: accuracy, precision, and recall.

7.4.2 SpliceTrans outperforms state-of-the-art in identifying splice sites of unseen species

Next we were curious to test the generalizability of the models in recognizing splice sites across different species. For this objective, we trained and validated each model with only human data and tested them on mouse and drosophila. In other words, this analysis tests the performance of the models on unseen species.

Figure 7.2 shows the performance of SpliceTrans, SpliceFinder, and SpliceRover trained with human data in identifying canonical and non-canonical splice junctions from mouse and drosophila test data. SpliceTrans obtains an F1-score up to 4% and 12% more than SpliceFinder in the identification of canonical (Figure 7.2(a)) and non-canonical (Figure 7.2(b)) splice junctions, respectively. On the other hand, F1-score obtained by SpliceTrans is up to 10% and 12% more than SpliceRover in the identification of canonical (Figure 7.2(c))



(a) human train, mouse test (can)

(b) human train, mouse test (non-can)

(c) human train, drosophila test (can)

(d) human train, drosophila test (non-can)

Figure 7.2: Performance (in percentage) obtained by SpliceTrans, SpliceRover and SpliceFinder trained with human data in the identification of canonical (can) and non-canonical (non-can) splice junctions from mouse and drosophila test data.

and non-canonical (Figure 7.2(d)) splice junctions, respectively. This test further affirms the robustness of SpliceTrans compared to the baselines.

7.4.3 SpliceTrans is more robust with imbalanced training data

We test the robustness of the neural network models by training the models on an imbalanced dataset. An imbalanced dataset is generated by increasing the ratio of negative to positive samples. This ratio of negative to positive samples is called the decoy rate. We increase the decoy rate from 3 to 9 in an interval of 2. Since accuracy, precision and recall are not appropriate performance metrics in an imbalanced dataset; we display only F1-score in this analysis.

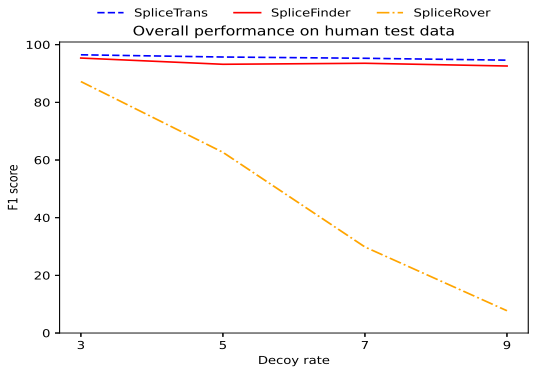
We train the models on the human dataset and test them on the human, mouse, and drosophila datasets. The positive and negative training samples are extracted from chromosome 1 (chr1) of the human dataset. We chose chromosome 1 since it is the largest chromosome in the human genome. Figure 7.3 displays the F1-scores obtained by SpliceTrans, SpliceFinder, and SpliceRover when trained on an imbalanced human dataset and tested on canonical and non-canonical splice sites from different species.

We observe that SpliceTrans consistently outperforms SpliceFinder and SpliceRover across different species even when the decoy rate increases. Further, we observe that the performance of SpliceRover significantly reduces by 80% as the decoy rate increases from 3 to 9. In particular, the performance of SpliceRover reduces to less than 50% when the decoy rate increases beyond 5. On the contrary, the performance of SpliceFinder and SpliceTrans reduces up to 10% and 6%, respectively. Therefore, we can conclude that SpliceTrans is the most robust model across several species to identify canonical and non-canonical splice sites.

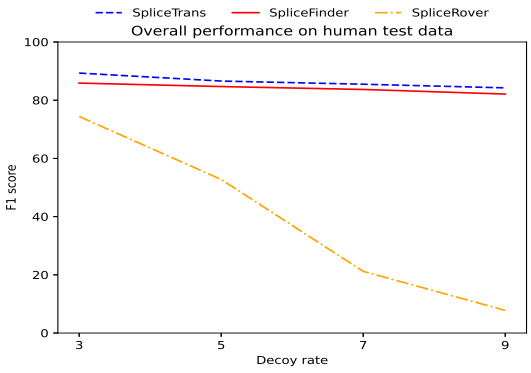
7.4.4 SpliceTrans outperforms state-of-the-art in identifying splice sites of partially annotated species

We were curious to evaluate whether the performance of the models can improve on augmenting the training data from one species with training data from another. If a model performs better with such an augmented dataset, it can then be applicable to annotate partially studied species using data from extensively annotated species. With this purpose, we extract training samples from chromosome 1 for the human and mouse training data. The training samples for drosophila were extracted from chromosome 3R. We choose these chromosomes since they are the largest chromosomes in their respective genome sequence data.

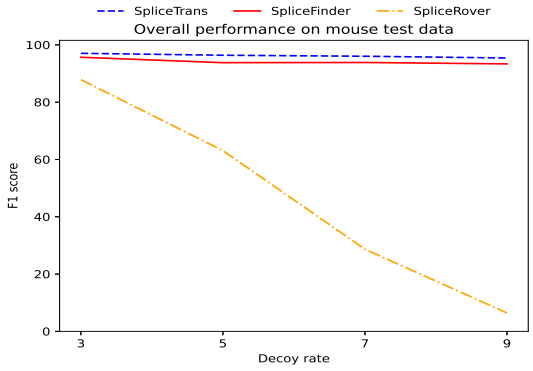
Figure 7.4(a) (Figure 7.4(b)) shows the performance of SpliceTrans, SpliceFinder, and SpliceRover in the identification of canonical and non-canonical splice sites when trained with only mouse (drosophila) data and mouse (drosophila) data combined with human data. We observe that all three models display improvement in the F1-score when trained using



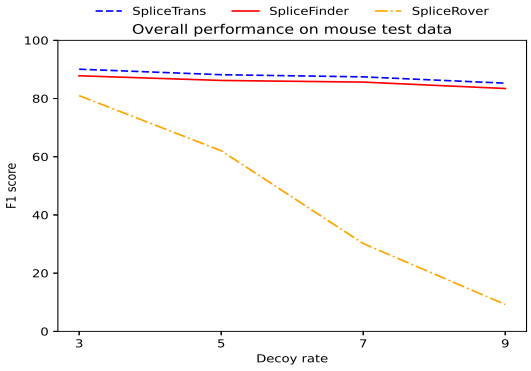
(a) human train, human test (can)



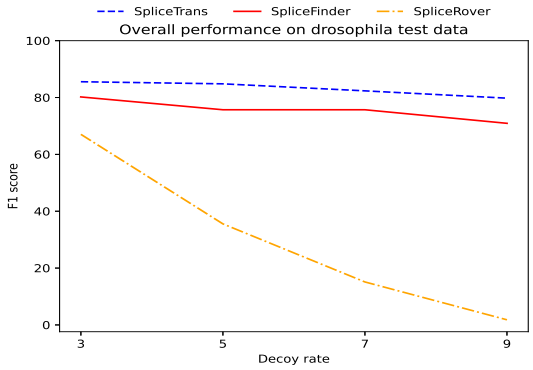
(b) human train, human test (non-can)



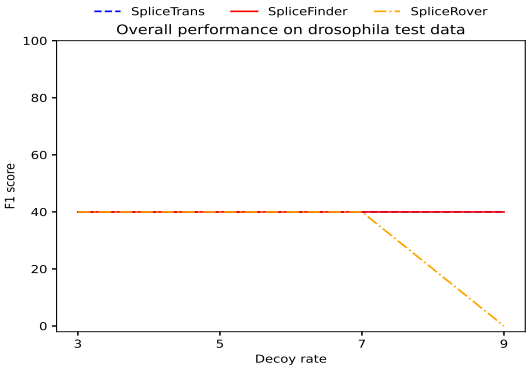
(c) human train, mouse test (can)



(d) human train, mouse test (non-can)

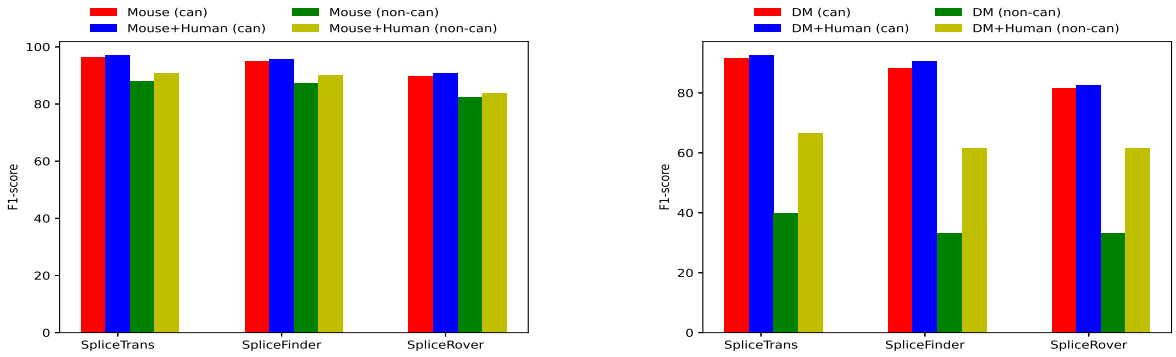


(e) human train, drosophila test (can)



(f) human train, drosophila test (non-can)

Figure 7.3: F1-score (in percentage) obtained by SpliceTrans, SpliceFinder and SpliceRover in the identification of splice junctions with imbalanced training data.



(a) mouse and human data augmented

(b) drosophila and human data augmented

Figure 7.4: F1-score (in percentage) obtained by SpliceTrans, SpliceFinder and SpliceRover in the identification of splice junctions with single species and multi-species training data.

data from two species. This improvement is observed in the case of both canonical and non-canonical splice site identification.

Hence, we can infer that combining the training data of one species with samples from another extensively annotated species can improve the model's performance. This performance improvement can be attributed to the increase in the number and variation of training data when multiple species are used. This analogy motivates annotating poorly studied or newly annotated species by training a model with data from another extensively annotated species.

In canonical splice sites, all three models show an improvement of approximately 1% when human data is added for training the model instead of using only mouse or drosophila data. However, in non-canonical splice sites, the improvement is up to 28% when human data is added to the training dataset. The more significant improvement in identifying non-canonical splice sites can be attributed to the fact that canonical splice site motifs are primarily similar across all eukaryotes [2, 127]. On the other hand, non-canonical splice sites show a wider range of variations and frequencies of occurrence. Therefore higher variation in the training data assists the models to recognize more unseen non-canonical splice sites.

Furthermore, we observe that SpliceTrans performs better than SpliceFinder and SpliceRover in both the scenario when one or two species are used to train the model. SpliceTrans shows an improvement of up to 7% compared to SpliceFinder when only mouse or drosophila data is used for training the models. The performance of SpliceTrans improves up to 5% compared to SpliceFinder when human data is also used for training. The improvement of SpliceTrans compared to SpliceRover goes up to 10%.

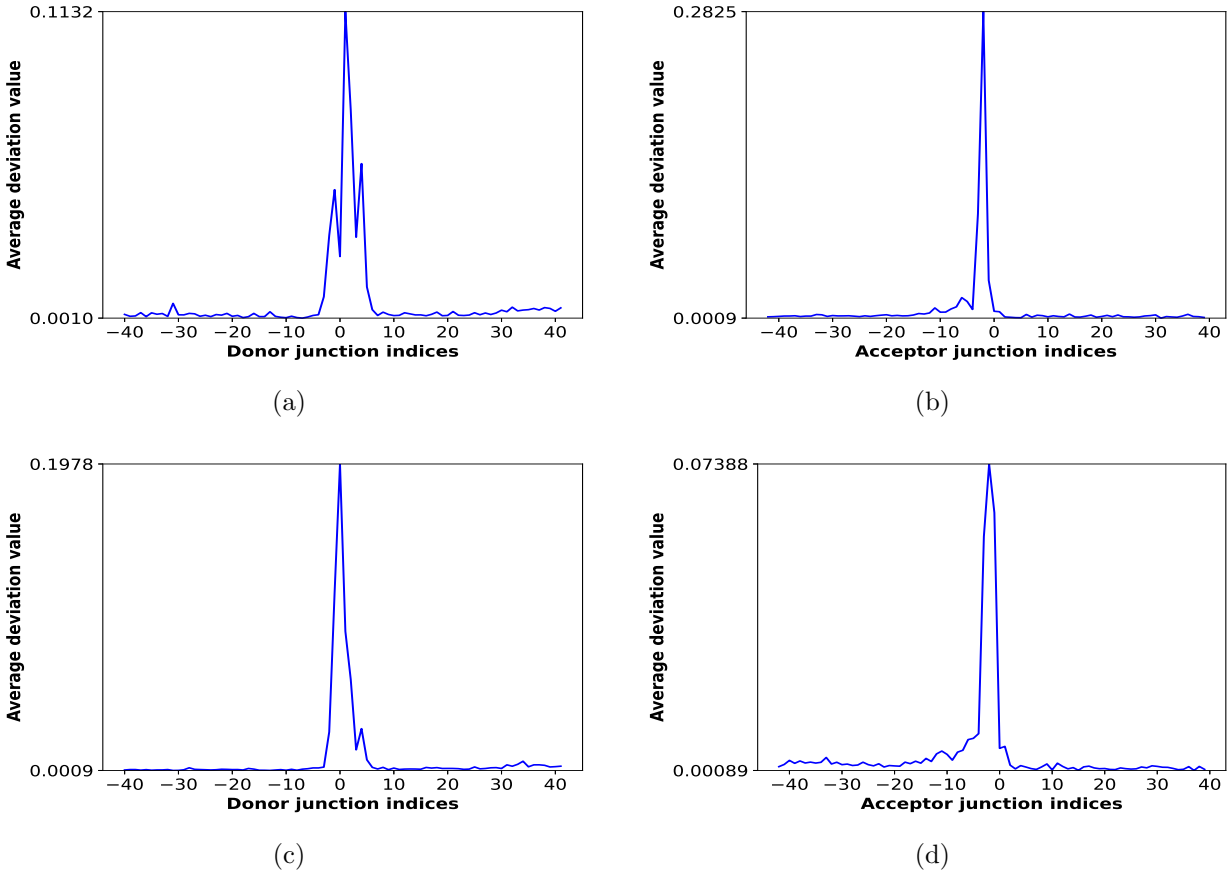


Figure 7.5: The importance of sequence positions. The average deviation value per position obtained by integrated gradients for non-canonical (a) donor junction and (b) acceptor junction in mouse+human model; non-canonical (c) donor junction and (d) acceptor junction in mouse model.

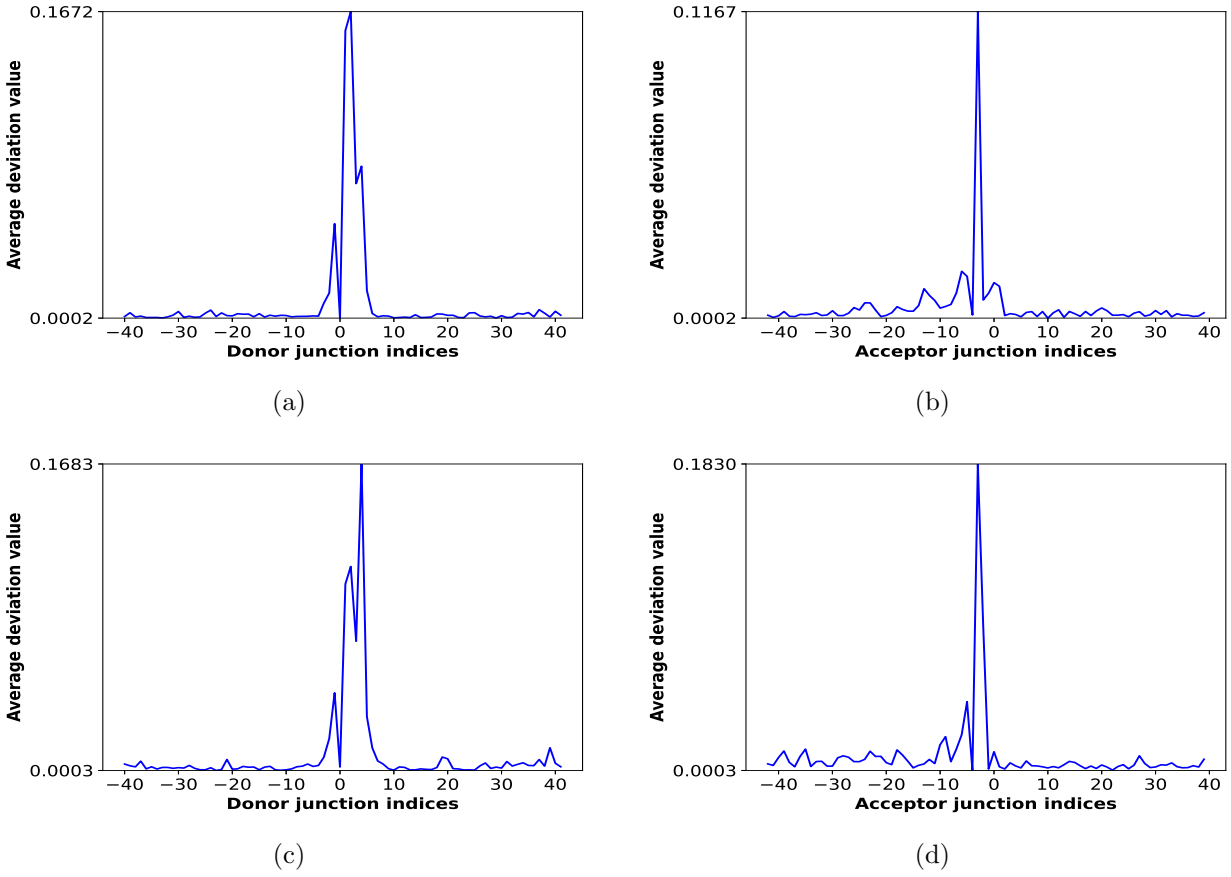


Figure 7.6: The importance of sequence positions. The average deviation value per position obtained by integrated gradients for non-canonical (a) donor junction and (b) acceptor junction in drosophila+human model; non-canonical (c) donor junction and (d) acceptor junction in drosophila model.

7.4.5 SpliceTrans captures significant sequence positions

As observed in Figure 7.4, combining data from two species for training the models significantly improves the performance in non-canonical splice sites. Therefore, we extract the features captured by SpliceTrans in both the cases from the non-canonical mouse and drosophila test data and compare the findings from both.

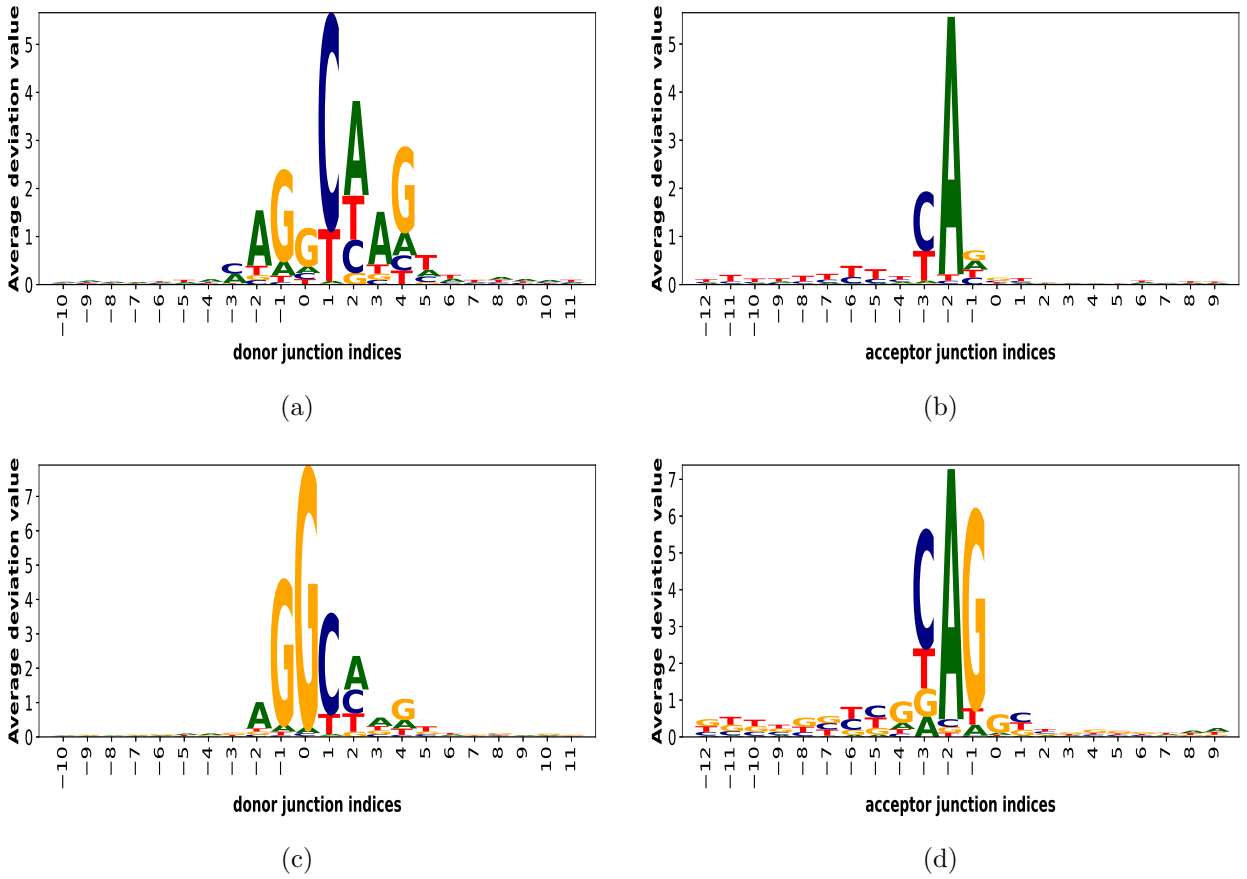


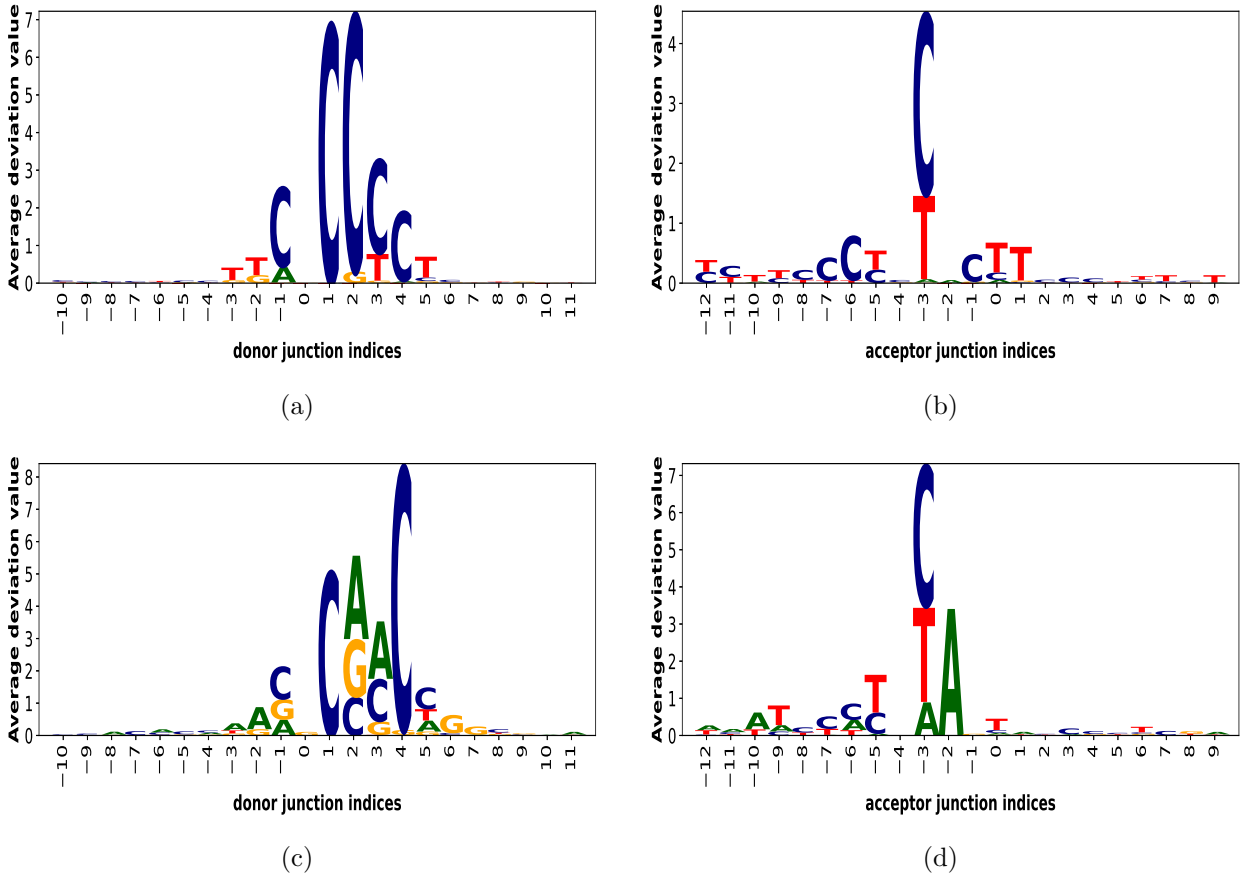
Figure 7.7: Sequence motifs at splice junctions. The average deviation value per position per nucleotide is shown by integrated gradients for non-canonical (a) donor junctions and (b) acceptor junctions in mouse+human model; non-canonical (c) donor junctions and (d) acceptor junctions in mouse model.

7.4.5.1 Significant sequence positions captured in mouse

Figure 7.5(a) and Figure 7.5(c) display the importance of sequence positions captured for non-canonical donor sites by SpliceTrans when trained by mouse+human and mouse data, respectively. We observe that the mouse model captures only the splice site and its downstream region as significant. In contrast, the mouse+human model captures the upstream region of the donor site as significant as well. Since the mouse and human splice site consensus at donor and acceptor sites are conserved [2, 105], we can derive that the mouse+human model

captures the extended upstream region of the donor site consensus 9-mer $[AC]AGGTRAGT$ whereas the mouse model does not.

The significant positions captured by both the models for the acceptor junction (Figure 7.5(b) and Figure 7.5(d)) seem to be identical. However, the importance of sequence positions beyond -15 nt upstream of the acceptor site appears smoother when mouse+human data (Figure 7.5(b)) is used. Beyond -15 nt upstream of the acceptor junction is the position where the polypyrimidine tract ends.



(a) (b)

(c) (d)

Figure 7.8: The importance of nucleotides per sequence position. The average deviation value per nucleotide per position is shown by integrated gradients for non-canonical (a) donor junction and (b) acceptor junction in drosophila+human model; non-canonical (c) donor junction and (d) acceptor junction in drosophila model.

7.4.5.2 Significant sequence positions captured in drosophila

The importance of sequence positions at the donor sites of drosophila is depicted in Figure 7.6(a) and Figure 7.6(c) for drosophila+human and drosophila model, respectively. We observe that the drosophila+human model gives maximum importance to position [0] at the

donor junction compared to the extended upstream and downstream region. On the contrary, the drosophila model gives more importance to the upstream region than the donor junction.

As known from the literature, the consensus dimer at the junctions differentiate non-canonical splice sites from their canonical counterparts. Intuitively, the importance given by the drosophila+human model (Figure 7.6(a)) abides by this rule thus making more sense than the drosophila (Figure 7.6(c)) model. In the case of acceptor junctions, drosophila+human (Figure 7.6(b)) model captures downstream region of the junction which the drosophila ((Figure 7.6(d))) model does not. Furthermore, the region captured beyond the PY-tract appears smoother in the drosophila+human model.

7.4.6 SpliceTrans captures splice junction consensus

We observe that SpliceTrans captures donor and acceptor splice site motifs in the case of both mouse+human (Figure 7.7(a) and Figure 7.7(b)) and mouse (Figure 7.7(c) and Figure 7.7(d)) training data. However, on a closer observation, we see that the donor site motif obtained from mouse+human training data (Figure 7.7(a)) covers more extended consensus compared to that of the mouse training data (Figure 7.7(c)).

Additionally, The mouse+human model gives highest importance to position [0] which corresponds to the most common non-canonical donor site consensus [GC]. The PY-tract captured by mouse+human model (Figure 7.7(b)) is less noisy compared to that of the mouse model (Figure 7.7(d)). Smoothening of the PY-tract is also seen in the drosophila+human model (Figure 7.8(b)) compared to drosophila model (Figure 7.8(d)).

7.5 Chapter Summary

We apply a BLSTM model named SpliceTrans that outperforms the state-of-the-art models in the task of identifying canonical and non-canonical splice junctions in human, mouse, and drosophila datasets. SpliceTrans also outperforms the baselines in identifying splice sites from species on which it was not trained. SpliceTrans also attains the highest F1-score when trained with an imbalanced dataset.

We observe an improvement in the models' performance when data from multiple species are used for training. SpliceTrans performs better than the baselines with such augmented training data as well. Therefore, SpliceTrans can be used for annotating species that are either newly or poorly annotated using training data from extensively annotated species.

We further extracted some relevant biological knowledge learnt by SpliceTrans through the application of integrated gradients. The knowledge thus obtained is validated with the existing literature. We also compare the features extracted when the model is trained with single-species and multiple-species training data. We observe that SpliceTrans extracts more

specific features when trained using data from more than one species.

"Nature holds the key to our aesthetic, intellectual, cognitive and even spiritual satisfaction."

Edward Osborne Wilson (1929)
American biologist

8

Conclusion and Future Directions

This thesis explores the neural models for analyzing the splicing mechanism that contributes to the transcript and protein diversity in eukaryotes. Splicing is regulated by regulatory instructions present in the genome sequence in the form of sequence patterns. Therefore, identification of these regulatory instructions/ signals are required to identify the splice sites accurately.

The traditional machine learning based techniques for identifying splice sites are mostly dependent on extraction and selection of functional genomic features for training the model. Such feature sets are neither optimal nor exhaustive. This led to the application of deep learning approaches which identify the splice sites and splicing signals from the genome sequences de novo. Although such models show promising performances, they function like black boxes making it difficult to extract and interpret the learnt features. Furthermore, the existing models primarily focus on identifying only canonical splice sites from a single species. This thesis fills up the research gaps mentioned above to understand the splicing phenomenon better.

8.1 Conclusions

In this thesis, we address the issues mentioned above by dividing the problems into the following objectives. Our objectives can be broadly divided into the following stages:

1. Hand-crafted features for training neural models are not optimal. Such feature sets can adversely affect the performance of the model. We propose two models (SpliceVec-g and SpliceVec-sp) based on widely used distributed representation models in natural language processing, namely word2vec and doc2vec. These models identify splice sites by learning features de novo from genome sequences. Here, the need for hand-crafted feature engineering has been eliminated to a great extent. SpliceVec is invariant to canonical and non-canonical splice junctions. The proposed model is consistent in its performance even with a reduced dataset and class-imbalanced dataset. We observe

that the inclusion of the entire intronic region in the input improves the model's performance significantly.

2. Feature extraction is difficult in representation models like SpliceVec due to limitations in intuitively explaining the embedding space. Therefore, we utilize a BLSTM based model, named SpliceVisuL, to extract the biological features learnt by the model de novo from genome sequences. We extract the learnt features through the application of several widely used visualization techniques. Furthermore, we compare the performance of the visualization techniques in identifying the known splicing features.

   We divided the visualization techniques into two categories: perturbation based and back-propagation based. We redesign the visualization techniques to be capable of comprehending genome sequences as inputs. We infer relevant biological information learnt by the model for both canonical and non-canonical splicing events and validate with the existing knowledge from the literature. We further compare and discuss the ability of the visualization techniques in the inference of known splicing features. Results indicate that the visualization techniques produce comparable performances for branchpoint detection. However, in the case of canonical donor and acceptor junction motifs, perturbation based visualizations perform better than back-propagation based visualizations and vice-versa for non-canonical motifs.

3. There is evidence in the literature suggesting that the signals regulating canonical and non-canonical splicing are possibly different from one another. This was also inferred in the previous objective. Therefore, we propose a BLSTM model named SpliceViNCI, which seeks to attain optimal performance in identifying non-canonical splicing in particular. The architecture of SpliceViNCI is similar to SpliceVisuL except that the attention layer is removed from SpliceViNCI since it failed to extract relevant features in the previous work.

   We also extract relevant features specific to non-canonical splicing. We use a perturbation based visualization cased occlusion and a back-propagation based visualization called integrated gradients to identify the non-canonical splicing features. Integrated gradient extracts features that comprise contiguous nucleotides, whereas occlusion extracts features that are individual nucleotides distributed across the sequence. Occlusion is helpful in the extraction of features dispersed far within the intronic regions.

4. Finally, we propose SpliceTrans to assess the BLSTM model's performance to identify splice sites from human, mouse, and drosophila melanogaster species with comparable accuracy. We also test the performance of SpliceTrans in identifying splice sites from species that were not used during the training phase. We observe that models trained with data from more than one species perform better, especially in identifying non-canonical splice sites. This suggests that the model can annotate poorly studied

species using data from extensively annotated species.

We also extract the non-canonical features captured by the model from the three species. We apply only back-propagation based visualization called integrated gradients because the previous two works suggested that back-propagation based visualization works better than perturbation based visualization in the case of non-canonical splicing features. Occlusion is helpful when more extended flanking regions are studied at the splice sites. We choose integrated gradients for feature extraction since this model extracts features from a shorter context (40 nt) at non-canonical splice junctions. We observe that SpliceTrans extracts more specific features when trained using data from more than one species.

8.2 Future Directions

The research done in this thesis can be extended in the following directions:

1. **Annotation of all splice sites from a genomic region:** The work done in this thesis identifies whether an input sequence contains a splice site or not. This task can be extended to find all the splice sites from a genomic region using a sliding window that captures the potential splice site at the center with the upstream and downstream flanking region.

   Similar work has been done by Jaganathan et al. in [59] where they used a sliding window to annotate splice sites in a genomic region using a dilated convolutional layers. However, they considered only canonical splice sites for training the model. Training a model to identify canonical and non-canonical splice sites from a given genomic region will be a more robust tool. Furthermore, a sliding window identifying all the splice sites in a genomic region modifies the splice site classification task to the splice site annotation task, which is one of the ultimate goals of studying the splicing mechanism.

2. **Annotation of other gene structure components :** A complete annotation system can be developed which identifies not only splice sites but also other components of the gene structure like transcription start sites and transcription termination sites. This is important because the overall cell mechanism is governed by regulatory signals present in different regions of the gene which regulate different cell activities. It is therefore important to annotate different regions of the gene for a comprehensive understanding of the cell mechanism.

   To annotate various gene structure components, a single model can be trained with labeled genomic sequences comprising various structural components. This can be

considered a multi-class classification task. Alternatively, various models can be separately trained for each of the components, and subsequently, an ensemble model can be applied to annotate the structural components in the genome sequences.

3. **Prediction of other cell variables:** The potential of deep learning models can also be explored in extracting features that govern other cell variables like polyadenylation and transcription site selection. Similar to splicing, the other cell variables also form a crucial part of gene expression. Therefore, a more comprehensive study to identify the governing features of different cell variables will help better understand the related cell activity and the diseases associated with their malfunctioning.

   We can apply similar models and curate the dataset using similar methodologies applied in this thesis for prediction and feature extraction of other cell variables. Furthermore, we can explore the recently proposed models in NLP like transformers with additional parallelism of a multi-head attention mechanism.

Bibliography

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M. et al. [2016], 'Tensorflow: Large-scale machine learning on heterogeneous distributed systems', *arXiv preprint arXiv:1603.04467* .

[2] Abril, J. F., Castelo, R. and Guigó, R. [2005], 'Comparison of splice sites in mammals and chicken', *Genome research* **15**(1), 111–119.

[3] Akerman, M. and Mandel-Gutfreund, Y. [2007], 'Does distance matter? variations in alternative 3 splicing regulation', *Nucleic acids research* **35**(16), 5487–5498.

[4] Albaradei, S., Magana-Mora, A., Thafar, M., Uludag, M., Bajic, V. B., Gojobori, T., Essack, M. and Jankovic, B. R. [2020], 'Splice2Deep: An ensemble of deep convolutional neural networks for improved splice site prediction in genomic DNA', *Gene: X* **5**, 100035.

[5] Alberts, B. [2008], 'Molecular biology of the cell'.

[6] 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* .

[7] Ancona, M., Ceolini, E., Oztireli, C. and Gross, M. [2018], Towards better understanding of gradient-based attribution methods for deep neural networks, *in* '6th International Conference on Learning Representations (ICLR 2018)'.

[8] Angermueller, C., Lee, H. J., Reik, W. and Stegle, O. [2017], 'DeepCpG: accurate prediction of single-cell DNA methylation states using deep learning', *Genome biology* **18**(1), 67.

[9] Asgari, E. and Mofrad, M. R. [2015], 'Continuous distributed representation of biological sequences for deep proteomics and genomics', *PloS one* **10**(11), e0141287.

[10] Asgari, E., Poerner, N., McHardy, A. and Mofrad, M. [2019], 'Deepprime2sec: Deep learning for protein secondary structure prediction from the primary sequences', *bioRxiv* p. 705426.

[11] Au, K. F., Jiang, H., Lin, L., Xing, Y. and Wong, W. H. [2010], 'Detection of splice junctions from paired-end RNA-seq data by splicemap', *Nucleic acids research* **38**(14), 4570–4578.

[12] Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K. and MÃžller, K.-R. [2010], 'How to explain individual classification decisions', *Journal of Machine Learning Research* **11**(Jun), 1803–1831.

[13] Bahdanau, D., Cho, K. and Bengio, Y. [2015], 'Neural machine translation by jointly learning to align and translate', *ICLR* .

[14] Bao, S., Moakley, D. F. and Zhang, C. [2019], 'The splicing code goes deep', *Cell* **176**(3), 414–416.

[15] Barash, Y., Calarco, J. A., Gao, W., Pan, Q., Wang, X., Shai, O., Blencowe, B. J. and Frey, B. J. [2010], 'Deciphering the splicing code', *Nature* **465**(7294), 53–59.

[16] Bari, A., Reaz, M. R. and Jeong, B.-S. [2014], 'Effective DNA encoding for splice site prediction using SVM', *MATCH Commun. Math. Comput. Chem* **71**, 241–258.

[17] Baten, A. K., Chang, B. C., Halgamuge, S. K. and Li, J. [2006], 'Splice site identification using probabilistic parameters and SVM classification', *BMC bioinformatics* **7**(5), 1.

[18] Berget, S. M. [1995], 'Exon recognition in vertebrate splicing', *Journal of biological Chemistry* **270**(6), 2411–2414.

[19] Binder, A., Montavon, G., Lapuschkin, S., Müller, K.-R. and Samek, W. [2016], Layer-wise relevance propagation for neural networks with local renormalization layers, *in* 'International Conference on Artificial Neural Networks', Springer, pp. 63–71.

[20] Boon, K.-L., Grainger, R. J., Ehsani, P., Barrass, J. D., Auchynnikava, T., Inglehearn, C. F. and Beggs, J. D. [2007], 'prp8 mutations that cause human retinitis pigmentosa lead to a u5 snrnp maturation defect in yeast', *Nature structural & molecular biology* **14**(11), 1077–1083.

[21] Bretschneider, H., Gandhi, S., Deshwar, A. G., Zuberi, K. and Frey, B. J. [2018], 'COSSMO: predicting competitive alternative splice site selection using deep learning', *Bioinformatics* **34**(13), i429–i437.

[22] Brunak, S., Engelbrecht, J. and Knudsen, S. [1991], 'Prediction of human mRNA donor and acceptor sites from the DNA sequence', *Journal of molecular biology* **220**(1), 49–65.

[23] 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.

[24] Burset, M., Seledtsov, I. and Solovyev, V. [2000], 'Analysis of canonical and non-canonical splice sites in mammalian genomes', *Nucleic acids research* **28**(21), 4364–4375.

[25] Cai, T. and Peng, Q. [2005], Predicting the splice sites in DNA sequences using neural network based on complementary encoding method, *in* '2005 International Conference on Neural Networks and Brain', Vol. 1, IEEE, pp. 473–476.

[26] Cartegni, L. and Krainer, A. R. [2002], 'Disruption of an SF2/ASF-dependent exonic splicing enhancer in SMN2 causes spinal muscular atrophy in the absence of SMN1', *Nature genetics* **30**(4), 377–384.

[27] Chao, H., Mansfield, S. G., Bartel, R. C., Hiriyanna, S., Mitchell, L. G., Garcia-Blanco, M. A. and Walsh, C. E. [2003], 'Phenotype correction of hemophilia a mice by spliceosome-mediated RNA trans-splicing', *Nature medicine* **9**(8), 1015–1019.

[28] Chen, T.-M., Lu, C.-C. and Li, W.-H. [2005], 'Prediction of splice sites with dependency graphs and their expanded bayesian networks', *Bioinformatics* **21**(4), 471–482.

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

[30] Cianferoni, A. and Spergel, J. [2014], 'The importance of TSLP in allergic disease and its role as a potential therapeutic target', *Expert review of clinical immunology* **10**(11), 1463–1474.

[31] Corvelo, A., Hallegger, M., Smith, C. W. and Eyras, E. [2010], 'Genome-wide association between branch point properties and alternative splicing', *PLoS computational biology* **6**(11), e1001016.

[32] de Klerk, E. and AC't Hoen, P. [2015], 'Alternative mRNA transcription, processing, and translation: insights from rna sequencing', *Trends in Genetics* **31**(3), 128–139.

[33] Degroeve, S., De Baets, B., Van de Peer, Y. and Rouzé, P. [2002], 'Feature subset selection for splice site prediction', *Bioinformatics* **18**(suppl 2), S75–S83.

[34] Degroeve, S., Saeys, Y., De Baets, B., Rouzé, P. and Van de Peer, Y. [2004], 'SpliceMachine: predicting splice sites from high-dimensional local context representations', *Bioinformatics* **21**(8), 1332–1338.

[35] Desmet, F.-O., Hamroun, D., Lalande, M., Collod-Béroud, G., Claustres, M. and Béroud, C. [2009], 'Human splicing finder: an online bioinformatics tool to predict splicing signals', *Nucleic acids research* **37**(9), e67–e67.

[36] Du, M., Liu, N. and Hu, X. [2018], 'Techniques for interpretable machine learning', *arXiv preprint arXiv:1808.00033* .

[37] Dutta, A., Dalmia, A., Athul, R., Singh, K. K. and Anand, A. [2019], 'Using the chou's 5-steps rule to predict splice junctions with interpretable bidirectional long short-term memory networks', *Computers in Biology and Medicine* .

[38] Dutta, A., Dubey, T., Singh, K. K. and Anand, A. [2018], 'SpliceVec: distributed feature representations for splice junction prediction', *Computational biology and chemistry* **74**, 434–441.

[39] Dutta, A., Singh, K. K. and Anand, A. [2020], 'SpliceViNCI: Visualizing the splicing of non-canonical introns through recurrent neural networks', *BioRxiv* .

[40] Gao, M., Zhou, H. and Skolnick, J. [2019], 'DESTINI: A deep-learning approach to contact-driven protein structure prediction', *Scientific reports* **9**(1), 3514.

[41] Gers, F. A., Schmidhuber, J. and Cummins, F. [1999], 'Learning to forget: Continual prediction with LSTM'.

[42] Goel, N., Singh, S. and Aseri, T. C. [2013], 'A comparative analysis of soft computing techniques for gene prediction', *Analytical biochemistry* **438**(1), 14–21.

[43] Goel, N., Singh, S. and Aseri, T. C. [2015], 'An improved method for splice site prediction in DNA sequences using support vector machines', *Procedia Computer Science* **57**, 358–367.

[44] Grant, G. R., Farkas, M. H., Pizarro, A. D., Lahens, N. F., Schug, J., Brunk, B. P., Stoeckert, C. J., Hogenesch, J. B. and Pierce, E. A. [2011], 'Comparative analysis of rna-seq alignment algorithms and the rna-seq unified mapper (RUM)', *Bioinformatics* **27**(18), 2518–2528.

[45] Graves, A. and Schmidhuber, J. [2005], 'Framewise phoneme classification with bidirectional LSTM and other neural network architectures', *Neural Networks* **18**(5-6), 602–610.

[46] Guler, R., Roy, S., Suzuki, H. and Brombacher, F. [2015], 'Targeting batf2 for infectious diseases and cancer', *Oncotarget* **6**(29), 26575.

[47] Harrow, J., Frankish, A., Gonzalez, J. M., Tapanari, E., Diekhans, M., Kokocinski, F., Aken, B. L., Barrell, D., Zadissa, A., Searle, S. et al. [2012], 'GENCODE: the reference human genome annotation for the ENCODE project', *Genome research* **22**(9), 1760–1774.

[48] Hatzigeorgiou, A., Mache, N. and Reczko, M. [1996], Functional site prediction on the DNA sequence by artificial neural networks, *in* 'Intelligence and Systems, 1996., IEEE International Joint Symposia on', IEEE, pp. 12–17.

[49] Hill, S. T., Kuintzle, R., Teegarden, A., Merrill III, E., Danaee, P. and Hendrix, D. A. [2018], 'A deep recurrent neural network discovers complex biological rules to decipher RNA protein-coding potential', *Nucleic acids research* **46**(16), 8105–8113.

[50] Ho, L. S. and Rajapakse, J. C. [2003*a*], 'Splice site detection with a higher-order markov model implemented on a neural network', *Genome Informatics* **14**, 64–72.

[51] Ho, L. S. and Rajapakse, J. C. [2003*b*], 'Splice site detection with a higher-order markov model implemented on a neural network', *Genome Informatics* **14**, 64–72.

[52] Ho, L. S. and Rajapakse, J. C. [2003*c*], 'Splice site detection with a higher-order markov model implemented on a neural network', *Genome Informatics* **14**, 64–72.

[53] Hochreiter, S., Heusel, M. and Obermayer, K. [2007], 'Fast model-based protein homology detection without alignment', *Bioinformatics* **23**(14), 1728–1736.

[54] Holmes, R. S., VandeBerg, J. L. and Cox, L. A. [2011], 'Genomics and proteomics of vertebrate cholesterol ester lipase (LIPA) and cholesterol 25-hydroxylase (CH25H)', *3 Biotech* **1**(2), 99–109.

[55] Hou, J., Wu, T., Cao, R. and Cheng, J. [2019], 'Protein tertiary structure modeling driven by deep learning and contact distance prediction in CASP13', *Proteins: Structure, Function, and Bioinformatics* .

[56] Huang, J., Li, T., Chen, K. and Wu, J. [2006], 'An approach of encoding for prediction of splice sites using SVM', *Biochimie* **88**(7), 923–929.

[57] Hubbard, T., Barker, D., Birney, E., Cameron, G., Chen, Y., Clark, L., Cox, T., Cuff, J., Curwen, V., Down, T. et al. [2002], 'The ensembl genome database project', *Nucleic acids research* **30**(1), 38–41.

[58] Islamaj, R., Getoor, L. and Wilbur, W. J. [2006], A feature generation algorithm for sequences with application to splice-site prediction, *in* 'European Conference on Principles of Data Mining and Knowledge Discovery', Springer, pp. 553–560.

[59] Jaganathan, K., Panagiotopoulou, S. K., McRae, J. F., Darbandi, S. F., Knowles, D., Li, Y. I., Kosmicki, J. A., Arbelaez, J., Cui, W., Schwartz, G. B. et al. [2019], 'Predicting splicing from primary sequence with deep learning', *Cell* **176**(3), 535–548.

[60] Kádár, A., Chrupała, G. and Alishahi, A. [2017], 'Representation of linguistic form and function in recurrent neural networks', *Computational Linguistics* **43**(4), 761–780.

[61] Kalkatawi, M., Rangkuti, F., Schramm, M., Jankovic, B. R., Kamau, A., Chowdhary, R., Archer, J. A. and Bajic, V. B. [2011], 'Dragon polya spotter: predictor of poly (A) motifs within human genomic dna sequences', *Bioinformatics* **28**(1), 127–129.

[62] Katz, Y., Wang, E. T., Airoldi, E. M. and Burge, C. B. [2010], 'Analysis and design of RNA sequencing experiments for identifying isoform regulation', *Nature methods* **7**(12), 1009–1015.

[63] Kellis, M., Wold, B., Snyder, M. P., Bernstein, B. E., Kundaje, A., Marinov, G. K., Ward, L. D., Birney, E., Crawford, G. E., Dekker, J. et al. [2014], 'Defining functional DNA elements in the human genome', *Proceedings of the National Academy of Sciences* **111**(17), 6131–6138.

[64] Kimothi, D., Soni, A., Biyani, P. and Hogan, J. M. [2016], 'Distributed representations for biological sequence analysis', *arXiv preprint arXiv:1608.05949* .

[65] Kingma, D. and Ba, J. [2014], 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980* .

[66] Koller, U., Wally, V., Bauer, J. W. and Murauer, E. M. [2014], 'Considerations for a successful RNA trans-splicing repair of genetic disorders', *Molecular Therapy-Nucleic Acids* **3**.

[67] Kumar, S. and Bucher, P. [2016], 'Predicting transcription factor site occupancy using DNA sequence intrinsic and cell-type specific chromatin features', *BMC bioinformatics* **17**(1), 41.

[68] Kunkel, T. A. [1985], 'Rapid and efficient site-specific mutagenesis without phenotypic selection', *Proceedings of the National Academy of Sciences* **82**(2), 488–492.

[69] Lanchantin, J., Singh, R., Wang, B. and Qi, Y. [2017], Deep motif dashboard: Visualizing and understanding genomic sequences using deep neural networks, *in* 'PACIFIC SYMPOSIUM ON BIOCOMPUTING 2017', World Scientific, pp. 254–265.

[70] Lander, E. S. [2011], 'Initial impact of the sequencing of the human genome', *Nature* **470**(7333), 187–197.

[71] Lander, E. S., Linton, L. M., Birren, B., Nusbaum, C., Zody, M. C., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W. et al. [2001], 'Initial sequencing and analysis of the human genome'.

[72] Le, Q. and Mikolov, T. [2014], Distributed representations of sentences and documents, *in* 'Proceedings of the 31st International Conference on Machine Learning (ICML-14)', pp. 1188–1196.

[73] Lee, B., Baek, J., Park, S. and Yoon, S. [2016], deepTarget: end-to-end learning framework for microRNA target prediction using deep recurrent neural networks, *in* 'Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics', ACM, pp. 434–442.

[74] Lee, B., Lee, T., Na, B. and Yoon, S. [2015], 'DNA-level splice junction prediction using deep recurrent neural networks', *arXiv preprint arXiv:1512.05135* .

[75] Lee, T. and Yoon, S. [2015], Boosted categorical restricted boltzmann machine for computational prediction of splice junctions, *in* 'International Conference on Machine Learning', pp. 2483–2492.

[76] Leung, M. K., Delong, A., Alipanahi, B. and Frey, B. J. [2016], 'Machine learning in genomic medicine: A review of computational problems and data sets', *Proceedings of the IEEE* **104**(1), 176–197.

[77] 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), 121–129.

[78] Li, B. and Dewey, C. N. [2011], 'RSEM: accurate transcript quantification from RNA-seq data with or without a reference genome', *BMC bioinformatics* **12**(1), 323.

[79] Li, Y., Chen, C.-y., Kaye, A. M. and Wasserman, W. W. [2015], 'The identification of cis-regulatory elements: A review from a machine learning perspective', *Biosystems* **138**, 6–17.

[80] Licatalosi, D. D. and Darnell, R. B. [2006], 'Splicing Regulation in Neurologic Disease', *Neuron* **52**(1), 93–101.

[81] Liu, L., Ho, Y.-K. and Yau, S. [2007], 'Prediction of primate splice site using inhomogeneous markov chain and neural network', *DNA and cell biology* **26**(7), 477–483.

[82] López-Bigas, N., Audit, B., Ouzounis, C., Parra, G. and Guigó, R. [2005], 'Are splicing mutations the most frequent cause of hereditary disease?', *FEBS letters* **579**(9), 1900–1903.

[83] Lundberg, S. M. and Lee, S.-I. [2017], A unified approach to interpreting model predictions, *in* 'Advances in neural information processing systems', pp. 4765–4774.

[84] Maaten, L. v. d. and Hinton, G. [2008], 'Visualizing data using t-SNE', *Journal of Machine Learning Research* **9**(Nov), 2579–2605.

[85] Matlin, A. J., Clark, F. and Smith, C. W. [2005], 'Understanding alternative splicing: towards a cellular code', *Nature reviews Molecular cell biology* **6**(5), 386–398.

[86] McClellan, J. and King, M.-C. [2010], 'Genetic heterogeneity in human disease', *Cell* **141**(2), 210–217.

[87] Meher, P. K., Sahu, T. K., Rao, A. and Wahi, S. [2016], 'Identification of donor splice sites using support vector machine: a computational approach based on positional, compositional and dependency features', *Algorithms for Molecular Biology* **11**(1), 1.

[88] Mikolov, T., Chen, K., Corrado, G. and Dean, J. [2013], 'Efficient estimation of word representations in vector space', *arXiv preprint arXiv:1301.3781* .

[89] Min, S., Lee, B. and Yoon, S. [2017], 'Deep learning in bioinformatics', *Briefings in bioinformatics* **18**(5), 851–869.

[90] Mirzoyan, Z., Sollazzo, M., Allocca, M., Valenza, A. M., Grifoni, D. and Bellosta, P. [2019], 'Drosophila melanogaster: a model organism to study cancer', *Frontiers in genetics* **10**, 51.

[91] Moghimi, F., Shalmani, M. T. M., Sedigh, A. K. and Kia, M. [2013], 'Two new methods for dna splice site prediction based on neuro-fuzzy network and clustering', *Neural Computing and Applications* **23**(1), 407–414.

[92] Montavon, G., Lapuschkin, S., Binder, A., Samek, W. and Müller, K.-R. [2017], 'Explaining nonlinear classification decisions with deep taylor decomposition', *Pattern Recognition* **65**, 211–222.

[93] Mount, S. M. [2000], 'Genomic sequence, splicing, and gene annotation', *American journal of human genetics* **67**(4), 788.

[94] Murdoch, W. J., Liu, P. J. and Yu, B. [2018], Beyond word importance: Contextual decomposition to extract interactions from LSTMs, *in* 'International Conference on Learning Representations'.

[95] Murray, J. I., Voelker, R. B., Henscheid, K. L., Warf, M. B. and Berglund, J. A. [2008], 'Identification of motifs that function in the splicing of non-canonical introns', *Genome biology* **9**(6), R97.

[96] Nassa, T., Singh, S. and Goel, N. [2013], 'Splice site detection in DNA sequences using probabilistic neural network', *International Journal of Computer Applications* **76**(4).

[97] Ng, P. [2017], 'dna2vec: Consistent vector representations of variable-length k-mers', *arXiv preprint arXiv:1701.06279* .

[98] Nisha and Kumar, S. [2013], 'Neural network based systems for splice site detection: A review', *International Journal of Advanced Research in Computer Science and Software Engineering* **3**(7), 636–640.

[99] Noordewier, M. O., Towell, G. G. and Shavlik, J. W. [1991], Training knowledge-based neural networks to recognize genes in DNA sequences, *in* 'Advances in neural information processing systems', pp. 530–536.

[100] Park, S., Min, S., Choi, H.-S. and Yoon, S. [2017], Deep recurrent neural network-based identification of precursor micrornas, *in* 'Advances in Neural Information Processing Systems', pp. 2891–2900.

[101] 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.

[102] Piovesan, A., Caracausi, M., Ricci, M., Strippoli, P., Vitale, L. and Pelleri, M. C. [2015], 'Identification of minimal eukaryotic introns through genebase, a user-friendly tool for parsing the ncbi gene databank', *DNA Research* **22**(6), 495–503.

[103] Poerner, N., Schütze, H. and Roth, B. [2018], Evaluating neural network explanation methods using hybrid documents and morphological prediction, *in* '56th Annual Meeting of the Association for Computational Linguistics (ACL)'.

[104] Pradhan, S., Sengupta, M., Dutta, A., Bhattacharyya, K., Bag, S. K., Dutta, C. and Ray, K. [2010], 'Indian genetic disease database', *Nucleic acids research* **39**(suppl_1), D933–D938.

[105] Qu, W., Cingolani, P., Zeeberg, B. R. and Ruden, D. M. [2017], 'A bioinformatics-based alternative mRNA splicing code that may explain some disease mutations is conserved in animals', *Frontiers in genetics* **8**, 38.

[106] Reese, M. G., Eeckman, F. H., Kulp, D. and Haussler, D. [1997], 'Improved splice site detection in genie', *Journal of computational biology* **4**(3), 311–323.

[107] Rehurek, R. and Sojka, P. [2010], Software framework for topic modelling with large corpora, *in* 'In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks', Citeseer.

[108] Roser, M. and Ritchie, H. [2016], 'Burden of disease', *Our World in Data* . https://ourworldindata.org/burden-of-disease.

[109] Ryen, T., Eftes, T., Kjosmoen, T., Ruoff, P. et al. [2008], Splice site prediction using artificial neural networks, *in* 'International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics', Springer, pp. 102–113.

[110] Saeys, Y., Degroeve, S., Aeyels, D., Rouzé, P. and Van de Peer, Y. [2004], 'Feature selection for splice site prediction: a new method using EDA-based feature ranking', *BMC bioinformatics* **5**(1), 64.

[111] Saeys, Y., Degroeve, S., Aeyels, D., Van de Peer, Y. and Rouzé, P. [2002], 'Selecting relevant features for splice site prediction by estimation of distribution algorithms', *Proceedings of Benelearn 2002* pp. 64–71.

[112] Saeys, Y., Degroeve, S. and Van de Peer, Y. [2004], Digging into acceptor splice site prediction: an iterative feature selection approach, *in* 'European Conference on Principles of Data Mining and Knowledge Discovery', Springer, pp. 386–397.

[113] Schölkopf, B., Tsuda, K. and Vert, J.-P. [n.d.], 'Accurate splice site detection for caenorhabditis elegans'.

[114] Sharma, N., Sosnay, P. R., Ramalho, A. S., Douville, C., Franca, A., Gottschalk, L. B., Park, J., Lee, M., Vecchio-Pagan, B., Raraigh, K. S. et al. [2014], 'Experimental assessment of splicing variants using expression minigenes and comparison with in silico predictions', *Human mutation* **35**(10), 1249–1259.

[115] Shenasa, H. and Hertel, K. J. [2019], 'Combinatorial regulation of alternative splicing', *Biochimica et Biophysica Acta (BBA)-Gene Regulatory Mechanisms* .

[116] Shomron, N. and Levy, C. [2009], 'MicroRNA-biogenesis and pre-mRNA splicing crosstalk', *BioMed Research International* .

[117] Shrikumar, A., Greenside, P. and Kundaje, A. [2017], Learning important features through propagating activation differences, *in* 'International Conference on Machine Learning', pp. 3145–3153.

[118] Sibley, C. R., Blazquez, L. and Ule, J. [2016], 'Lessons from non-canonical splicing', *Nature Reviews Genetics* **17**(7), 407.

[119] Smilkov, D., Thorat, N., Kim, B., Viégas, F. and Wattenberg, M. [2017], 'Smoothgrad: removing noise by adding noise', *arXiv preprint arXiv:1706.03825* .

[120] Sonnenburg, S., Schweikert, G., Philips, P., Behr, J. and Rätsch, G. [2007], 'Accurate splice site prediction using support vector machines', *BMC bioinformatics* **8**(10), 1.

[121] Sorek, R. and Ast, G. [2003], 'Intronic sequences flanking alternatively spliced exons are conserved between human and mouse', *Genome Research* **13**(7), 1631–1637.

[122] Sundararajan, M., Taly, A. and Yan, Q. [2017], Axiomatic attribution for deep networks, *in* 'Proceedings of the 34th International Conference on Machine Learning-Volume 70', JMLR. org, pp. 3319–3328.

[123] Tazi, J., Bakkour, N. and Stamm, S. [2009], 'Alternative splicing and disease', *Biochimica et Biophysica Acta - Molecular Basis of Disease* **1792**(1), 14–26.
URL: *http://dx.doi.org/10.1016/j.bbadis.2008.09.017*

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

[125] Vigouroux, C. and Bonne, G. [2003], 'Laminopathies: One gene, two proteins, five diseases', *Nuclear Envelope Dynamics in Embryos and Somatic Cells* pp. 153–172.

[126] Wang, K., Singh, D., Zeng, Z., Coleman, S. J., Huang, Y., Savich, G. L., He, X., Mieczkowski, P., Grimm, S. A., Perou, C. M. et al. [2010], 'MapSplice: accurate mapping of RNA-seq reads for splice junction discovery', *Nucleic acids research* **38**(18), e178–e178.

[127] Wang, R., Wang, Z., Wang, J. and Li, S. [2019], 'SpliceFinder: ab initio prediction of splice sites using convolutional neural network', *BMC bioinformatics* **20**(23), 1–13.

[128] Wang, Z. and Burge, C. B. [2008], 'Splicing regulation: from a parts list of regulatory elements to an integrated splicing code', *Rna* **14**(5), 802–813.

[129] Watson, J. D. and Crick, F. H. [1953], 'Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid', *Nature* **171**(4356), 737–738.

[130] Wei, D., Zhang, H., Wei, Y. and Jiang, Q. [2013], 'A novel splice site prediction method using support vector machine', *J Comput Inform Syst* **920**, 8053–60.

[131] Wilkie, S. E., Vaclavik, V., Wu, H., Bujakowska, K., Chakarova, C. F., Bhattacharya, S. S., Warren, M. J. and Hunt, D. M. [2008], 'Disease mechanism for retinitis pigmentosa (RP11) caused by missense mutations in the splicing factor gene PRPF31'.

[132] Xiong, H. Y., Barash, Y. and Frey, B. J. [2011], 'Bayesian prediction of tissue-regulated splicing using RNA sequence and cellular context', **27**(18), 2554–2562.

[133] Yin, M. M. and Wang, J. T. [2001], 'Effective hidden markov models for detecting splicing junction sites in dna sequences', *Information Sciences* **139**(1), 139–163.

[134] Zeiler, M. D. and Fergus, R. [2014], Visualizing and understanding convolutional networks, *in* 'European conference on computer vision', Springer, pp. 818–833.

[135] Zhang, Q., Peng, Q., Li, K., Kang, X. and Li, J. [2009], Splice sites detection by combining markov and hidden markov model, *in* '2009 2nd International Conference on Biomedical Engineering and Informatics', IEEE, pp. 1–5.

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

[137] Zhang, Y., Chu, C.-H., Chen, Y., Zha, H. and Ji, X. [2006], 'Splice site prediction using support vector machines with a bayes kernel', *Expert Systems with Applications* **30**(1), 73–81.

[138] Zhang, Y., Liu, X., MacLeod, J. and Liu, J. [2018], 'Discerning novel splice junctions derived from RNA-seq alignment: a deep learning approach', *BMC genomics* **19**(1), 971.

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

[140] Zhou, Y., Kaiser, T., Monteiro, P., Zhang, X., Van der Goes, M. S., Wang, D., Barak, B., Zeng, M., Li, C., Lu, C. et al. [2016], 'Mice with shank3 mutations associated with ASD and schizophrenia display both shared and distinct defects', *Neuron* **89**(1), 147–162.

[141] Zhu, F., Nair, R. R., Fisher, E. M. and Cunningham, T. J. [2019], 'Humanising the mouse genome piece by piece', *Nature communications* **10**(1), 1–13.

[142] Zuallaert, J., Godin, F., Kim, M., Soete, A., Saeys, Y. and De Neve, W. [2018], 'SpliceRover: interpretable convolutional neural networks for improved splice site prediction', *Bioinformatics* **34**(24), 4180–4188.

Publications

Journals

- Dutta, Aparajita, Tushar Dubey, Kusum Kumari Singh, and Ashish Anand. "SpliceVec: distributed feature representations for splice junction prediction." Computational biology and chemistry 74 (2018): 434-441.

- Dutta, Aparajita, Aman Dalmia, R. Athul, Kusum Kumari Singh, and Ashish Anand. "Using the Chou's 5-steps rule to predict splice junctions with interpretable bidirectional long short-term memory networks." Computers in biology and medicine 116 (2020): 103558.

- Dutta, Aparajita, Kusum Kumari Singh, and Ashish Anand. "SpliceViNCI: Visualizing the splicing of non-canonical introns through recurrent neural networks." Journal of Bioinformatics and Computational Biology (2021): 2150014.

Conferences

- Dutta, Aparajita, Tushar Dubey, Kusum Kumari Singh, and Ashish Anand. "SpliceVec: distributed feature representations for splice junction prediction." APBC Japan 2018.

- Aparajita Dutta and Ashish Anand. Neural network models for analyzing the splicing cell variable from genome sequences. EMBO Symposium "Regulatory epigenomics: From large data to useful models", 2019, Chennai, India.

Manuscripts (Under review)

- Dutta, Aparajita, Kusum Kumari Singh, and Ashish Anand. "Deep learning models for identification of splice junctions across species." bioRxiv (2021).

Tools on GitHub

- Dutta, Aparajita, Tushar Dubey, Kusum Kumari Singh, and Ashish Anand. "SpliceVec: distributed feature representations for splice junction prediction". GitHub URL: https://github.com/aaiitggrp/SpliceVec-g , 2018.

- Dutta, Aparajita, Aman Dalmia, R. Athul, Kusum Kumari Singh, and Ashish Anand. "SpliceVisuL: Visualization of Bidirectional Long Short-term Memory Networks for Splice Junction Prediction". GitHub URL: https://github.com/aaiitggrp/SpliceVisuL , 2019.

Vitae

Aparajita Dutta joined the Ph.D. programme at the Department of Computer Science and Engineering (CSE) of Indian Institute of Technology (IIT) Guwahati, India, in July 2015. Before joining her Ph.D., she did her Bachelor of Technology degree from National Institute of Technology (NIT) Silchar and Master of Engineering degree from Jadavpur University. She has previously worked with Odessa Technologies in 2014-2015 as a Software Engineer in the Research and Development Team. Her job was centered around maintaining an existing leasing software (back end development) and its user interface (front end development). She has also worked as a graduate teaching assistant in NIT Nagaland (2011-2012), where she was the course instructor of C and Graph Theory courses in the department of CSE. She was a GATE fellowship recipient from MHRD, Govt. of India during her Masters and Ph.D. She received the NEC Scholarship during her B.Tech. from MDoNER, Govt. of India. She also received the Anundoram Borooah Award in the 10th board Examination, 2005 by Govt. of Assam for scoring distinction (85%) marks. Her research interests are Bioinformatics, Computational Biology, Machine Learning, and Deep Learning. She is also passionate about art, music, travel, and reading.

Contact Information

Email	:	d.aparajita@iitg.ac.in, aparajitanits@gmail.com
Web	:	https://www.iitg.ac.in/stud/d.aparajita/