UNIVERSIDADE DE LISBOA

FACULDADE DE CIÊNCIAS

Leveraging Deep Learning Models for Studying RNA Splicing in Health and Disease

"Documento Provisório"

Doutoramento em Informática

Pedro Santos Barbosa

Tese orientada por:

Professor Doutor Alcides Fonseca

Professora Doutora Maria Carmo-Fonseca

Documento especialmente elaborado para a obtenção do grau de doutor

2024

UNIVERSIDADE DE LISBOA

FACULDADE DE CIÊNCIAS

Leveraging Deep Learning Models for Studying RNA Splicing in Health and Disease

Doutoramento em Informática

Pedro Santos Barbosa

Tese orientada por:

Professor Doutor Alcides Fonseca

Professora Doutora Maria Carmo-Fonseca

This work was supported by FCT through PhD Scholarships under the refs. SFRH/BD/137062/2018 and COVID/BD/152918/2022, and High Performance Computing Projects with the refs. CPCA/A2/6009/2020, CPCA/A1/5613/2020, CPCA/A1/402869/2021, 2022.15800.CPCA.A1 and 2024.AIVLAB.00002.VISION, by the LASIGE Research Unit, ref. UIDB/00408/2020 and UIDP/00408/2020, by the RAP project, ref. EXPL/CCI-COM/1306/2021, and the CMU-Portugal project CAMELOT, ref. LISBOA-01-0247-FEDER-045915.

Documento especialmente elaborado para a obtenção do grau de doutor

2024

Acknowledgments

First off, I would like to thank my supervisor, Alcides Fonseca, for his friendship and guidance throughout this journey. He challenged me to step into unfamiliar territories, and his practical approach to problem-solving will certainly be useful in my future endeavors. I am thankful to Carmo-Fonseca for welcoming me into the lab several years ago. This opportunity allowed me to grow as a scientist, interact with talented researchers, and nurture my curiosity across diverse fields. A special acknowledgment goes to Rosina, who, although not an official supervisor, stepped in during the most challenging times. Her ability to provide clarity and serve as a bridge between the multidisciplinary aspects of my work was crucial. Her help in pushing the work forward is something for which I am deeply grateful.

My appreciation to all the generations of scientists I encountered at iMM. To the first batch of students (how naive we were!), with whom I shared enjoyable moments in the lab, over dinners, and during nights out - Ana Raposo, Catarina, Cláudia, Maria, Miguel, Pedro Prudêncio, and Tajda - I am thankful. To Teresinha, the best lab-colleague-housemate I could have asked for. To Marta Furtado, for the numerous scientific discussions and willingness to help, and to Rui Luís for our shared struggles, and many laughs (including our favorite bioinformatics memes). I extend my gratitude to the past and current members of Carmo's lab, too numerous to name individually. To the canteen group that came together for lunches and grew into good friendships, especially Sara, for her true support during the toughest times. To Helena, my multi-faceted bike partner, and Inês Mendes for her contagious commitment to open science and reproducibility.

Thanks to LASIGE folks, especially those from the sala 30 crew, namely Allan, Diana, Miguel, Rita, and the remarkable ladies Alexandra and Carla. These two patiently listened to my rants and occasional bad moods, always engaging in meaningful conversations and sharing similar perspectives on life. A special note of gratitude to Ruben for our philosophical discussions on life, science, and politics, and for his curiosity in learning biology from me. To current members of Alcides's group, Guilherme, for generously sharing his knowledge on many topics and his technical advice. To Leon, for his frankness and activism in fighting the good causes. And to Paulo and Catarina, for their support and presence despite the physical distance.

To my family and friends back in Nine (best town in the world, no contest), thank you for your support and understanding my absent times, especially my sister who has always been there when needed. And to the Matilha group, for being a constant presence in my life for over 15 years, sticking with me through every challenge.

A special acknowledgment to Beatriz, who has been by my side through my darkest times, never giving up on me. Her presence provided the emotional stability I needed the most. She has taught me the importance of self-care and that there is a fulfilling life beyond work. I would not have made it without her.

Lastly, I dedicate this thesis to my father, whose unwavering support, freedom to pursue my goals, and values have deeply influenced my journey. I miss you! Equally, I am eternally grateful to my mom for her resilience, strength, and unconditional love. I hope I have made you two proud.

Summary

Deep learning models have demonstrated remarkable potential across various domains, including biology. Despite this, scientists and clinicians face significant challenges when using these tools in practice. One major concern is that biological systems are inherently complex, raising doubts about whether even the most advanced models can fully capture their intricacies. Additionally, the "black box" nature of these million-parameter models poses a barrier to researchers seeking not only accurate predictions but also mechanistic understanding.

This thesis investigates the practical applications of deep learning models within research contexts. It specifically focuses on models trained on genomic sequences to predict RNA splicing, the domain of application for this work. The research addresses two key challenges: variant effect prediction, which serves as a practical application to assess model performance, and model interpretability, which aims to advance scientific understanding of the underlying mechanisms of RNA splicing.

We first addressed the problem of predicting the effects of variants that affect splicing in deep intronic regions, which are often ignored in genetic tests but now are recognized as important. Through a comprehensive evaluation of state-of-the-art computational models on curated datasets of disease-causing deep intronic variants, we revealed the strengths and limitations of these methods. In particular, we found that pure sequence-based deep learning models, like SpliceAI and Pangolin, were effective in variant prediction and that models combining SpliceAI predictions with additional features did not improve performance. Nevertheless, we showed that regardless of the model, there is still room for improvement, especially for variants disrupting splicing regulatory elements, which were often misclassified.

For model interpretability, we conducted an in-depth analysis of SpliceAI to uncover its learned representations of RNA splicing mechanisms. Through large-scale ablation experiments, we investigated SpliceAI's ability to study alternative splicing. Our findings showed that SpliceAI distinguishes between constitutive and alternatively spliced exons and uses RNA-binding protein motifs as features for its predictions. However, we also highlight some limitations and cautions for its use in such analyses.

We further explored model interpretability by developing strategies to study deep learning models and splicing locally, at the individual exon level. We combined genetic programming algorithms with domain-aware grammars to produce semantically-rich synthetic datasets, demonstrating their suitability for local explainable AI in genomics. Additionally, we highlighted the expressive power of grammars in enabling the design of in silico experiments, using the deep learning model as an oracle. These concepts were integrated into a software, designed for splicing-related experiments, assuming the deep learning model accurately reflects the biological processes involved.

In conclusion, this thesis demonstrated both the potential and current limitations of using deep learning models as research tools for studying RNA splicing. This work also contributed open-source software and placed a strong focus on reproducibility, ensuring that this research can be adopted and extended by the broader scientific community.

Keywords: Deep Learning, RNA Splicing, Variant Effect Prediction, Explainable AI, Genetic Programming

Resumo

Os modelos de *deep learning* têm mostrado um grande potencial em várias áreas, incluindo a biologia. No entanto, cientistas e clínicos enfrentam vários desafios na aplicação real destes modelos. Por exemplo, dada a elevada complexidade dos sistemas biológicos, é difícil perceber se estes modelos prevêm com qualidade todas as suas nuances. Além disso, a falta de transparência destes modelos, que normalmente são compostos por milhões de parâmetros, carrega desafios adicionais para os investigadores que procuram compreender os mecanismos subjacentes às suas previsões.

Esta tese investiga as aplicações práticas de modelos de *deep learning* no contexto da investigação científica. Em particular, foca-se em modelos treinados a partir de sequências genómicas para prever o *splicing* do RNA, o domínio de aplicação deste trabalho. A investigação aborda dois desafios principais: a previsão dos efeitos de variantes genéticas, que serve como uma aplicação prática para avaliar o desempenho dos modelos, e a interpretabilidade, que visa avançar o conhecimento científico sobre o *splicing* do RNA.

Em primeiro lugar, abordámos o problema de prever o impacto de variantes que afetam o *splicing* e se localizam em regiões intrónicas do genoma, regiões estas frequentemente ignoradas em testes genéticos mas agora reconhecidas como importantes. Para tal, fizemos uma curação manual de variantes intrónicas causadoras de doença e usámos estes dados para avaliar dezenas de modelos computacionais na sua capacidade de previsão, revelando as suas vantagens e limitações. Em particular, descobrimos que os modelos de *deep learning* baseados apenas em sequências, como o SpliceAI e Pangolin, foram eficazes na previsão de variantes. Modelos que combinam previsões do SpliceAI com atributos (*features*) adicionais não melhoraram o desempenho. No entanto, independentemente do modelo, mostrámos que ainda há espaço para melhorias nesta tarefa, especialmente em variantes que afetam a afinidade da ligação de elementos reguladores do *splicing*.

Relativamente à interpretabilidade dos modelos, conduzimos uma análise mais detalhada do SpliceAI para investigar as representações sobre mecanismos de splicing que o modelo aprendeu. Para tal, conduzimos estudos de perturbações ao modelo para perceber a capacidade do SpliceAI de estudar *splicing* alternativo. Os resultados mostraram que o modelo é capaz de distinguir

entre exões constitutivos e alternativos e que usa locais de ligação (motivos) de proteínas reguladoras do *splicing* como *features* nas suas previsões. No entanto, revelámos também algumas limitações do SpliceAI e considerações a ter no seu uso em análises semelhantes.

De seguida, desenvolvemos estratégias para estudar localmente modelos de *deep learning* que prevêm *splicing*, isto é, ao nível do exão individual. Combinámos algoritmos de programação genética com gramáticas direcionadas ao domínio para gerar dados sintéticos que cobrem o espaço semântico do modelo, demonstrando o seu potencial para serem usados em Inteligência Artificial explicável no contexto da genómica. Também destacámos o poder expressivo das gramáticas no design de experiências *in silico*, usando o modelo de deep learning como fonte the conhecimento. Estes conceitos foram integrados num software, com o objetivo de fazer experiências computacionais relacionadas com o *splicing*, assumindo que o modelo de *deep learning* reflete com precisão os processos biológicos envolvidos.

Esta tese demonstrou o potencial e as limitações atuais do uso de modelos de *deep learning* como ferramentas de investigação para o estudo do *splicing* do RNA. O trabalho resultou em software de código aberto e prioritizou a reprodutibilidade, permitindo assim à comunidade científica avançar a investigação na área tendo em conta os conceitos apresentados.

Palavras Chave: *Deep Learning*, *Splicing* do RNA, Previsão de Variantes Genéticas, IA Explicável, Programação Genética.

Resumo Alargado

Os modelos de *deep learning* estão a transformar a investigação na área da biologia. Modelos treinados a partir de sequências genómicas conseguem prever com alta precisão vários mecanismos moleculares que ocorrem nas células. No entanto, dada a sua complexidade, é um desafio dissecar o que aprendem e se tal pode ser utilizado para avançar o conhecimento científico nas ciências da vida. Este trabalho explora a possibilidade de utilizar inteligência artificial como um recurso para otimizar experiências laboratoriais ou gerar novas hipóteses científicas. Como domínio de aplicação, a tese foca-se no *splicing* do RNA, um mecanismo molecular essencial na regulação da expressão genética.

O *splicing* é um passo fundamental no processamento do RNA que consiste na remoção de sequências não codificantes - os intrões - e subsequente ligação das regiões codificantes - os exões - numa sequência contígua que serve de base para a tradução do RNA mensageiro em proteínas. O *splicing* é molecularmente complexo, regulado por centenas de proteínas que coordenam entre si as diferentes fases do processo, e que está longe de ser totalmente compreendido.

O primeiro objetivo foi compreender como é que estes modelos prevêem o impacto de variantes genéticas que influenciam o mecanismo do *splicing*. A relevância desta tarefa é grande, pois estima-se que até 50% das variantes genéticas causadoras de doença afetam o *splicing*. Se os modelos previrem com precisão estas variantes, podem ser usados na prática clínica para a prioritização de candidatos identificados em testes genéticos, podendo contribuir para a personalização da terapia oferecida ao paciente.

Para a execução deste objetivo, focámo-nos no levantamento dos modelos existentes adaptados a essa tarefa, e na curação de dados (variantes) que permitam a sua avaliação (*benchmarking*). Quanto à primeira vertente, identificámos diversos modelos passíveis de avaliação, sendo estes bastante heterogéneos no seu funcionamento e *input* que recebem. Neste sentido, implementámos uma ferramenta para processar variantes em formato VCF e criar o *input* correto para um conjunto de modelos. Além disso, desenvolvemos um software - VETA - que automatiza a comparação dos modelos na tarefa de avaliar o efeito de variantes genéticas.

Relativamente à curação de dados, debruçámo-nos em identificar variantes que se localizam em regiões intrónicas dos genes, pois estas são regiões tradicionalmente ignoradas nos testes

genéticos, mas que atualmente têm uma importância reconhecida na regulação do *splicing*. De forma a adicionar maior novidade ao trabalho, fizemos um esforço para agrupar as variantes identificadas de acordo com o mecanismo molecular afetado. Por exemplo, uma variante que cria um *splice donor* é diferente duma variante que destrói o sinal de *branchpoint*. Assim sendo, para além duma avaliação destes modelos de previsão a nível geral, executámos comparações específicas a cada tipo de variante, a um nível de resolução nunca antes realizado.

Os resultados da avaliação demonstraram que modelos de *deep learning* treinados apenas em sequências, como o SpliceAI e Pangolin, foram os mais eficazes na previsão do impacto de variantes de *splicing* intrónicas. Observámos ainda que a combinação de previsões do SpliceAI com atributos (*features*) adicionais não melhoraram o seu desempenho. No entanto, independentemente do modelo avaliado, demonstrámos que ainda existe margem para melhoria nesta tarefa, especialmente em variantes que afetam a afinidade da ligação de elementos reguladores do *splicing*. Além disso, revelámos o balanço entre o desempenho dos modelos e a sua interpretabilidade intrínseca. Modelos treinados com dados tabulares (p.e. SPiP e SQUIRLS) e com foco em interpretabilidade revelaram um desempenho pior comparativamente a redes neuronais convolucionais como o SpliceAI e Pangolin.

O segundo grande objetivo deste trabalho foi estudar a interpretabilidade de modelos vistos como caixas negras, em particular o SpliceAI. A estratégia mais comum para explicar as previsões destes modelos implica o uso de técnicas que analisam o gradiente do modelo em relação à sequência de *input*, atribuindo uma importância a cada nucleótido para explicar a sua previsão. No entanto, para extrair padrões recorrentes nas sequências e derivar representações de *features* mais abrangentes, são necessárias análises adicionais complexas e normalmente distantes de um investigador especializado no domínio.

Tendo em conta estas limitações, optámos numa primeira instância por realizar estudos de perturbações ao modelo em larga escala. Estas perturbações foram automatizadas numa pipeline denominada MutSplice e foram orientadas ao domínio do *splicing*. Especificamente, perturbámos sequências que sofrem *splicing* alternativo após o silenciamento de genes identificados como reguladores de *splicing*. Ao perturbar estas sequências nos locais de ligação destes reguladores, verificámos que o modelo é sensível à sua presença e que a magnitude e direção do efeito das perturbações replica parcialmente o conhecimento existente acerca da regulação do *splicing*, em particular nas famílias de reguladores de *splicing* SR e hnRNPs. No entanto, avaliámos os resultados com prudência, referindo as limitações técnicas e conceptuais da análise, em particular o facto do SpliceAI ser agnóstico ao tipo celular onde a sequência é prevista.

Aprofundando o trabalho neste tópico, focámo-nos no uso destes modelos para estudar o *splicing* ao nível individual de cada sequência, com o objetivo de abordar Inteligência Artificial Explicável a nível local. Em particular, propusemos um novo método para gerar dados sintéticos - neste caso, sequências - que são semanticamente ricos no espaço de previsão do SpliceAI. Ou seja, a partir duma sequência genómica real, geramos sequências sintéticas semelhantes

à original, mas que produzem previsões bastante diferentes. Assumindo que o modelo é uma fonte fiável de conhecimento sobre o *splicing*, é possível estudá-lo através das sequências sintéticas geradas, seja treinando modelos *surrogate* interpretáveis de raiz, ou analisando os padrões de *features* utilizados em diferentes valores da previsão do modelo. A metodologia é baseada em Programação Genética em que o espaço de procura para gerar as perturbações nas sequências é limitado por gramáticas ajustadas ao domínio do *splicing*. Estas gramáticas codificam perturbações a aplicar à sequência original e condicionam o tipo e a sua localização de acordo com regras expressivas sobre o mecanismo de *splicing*. Por exemplo, o processo de geração não deve incluir perturbações nas zonas de *splice sites* pois o impacto biológico das mesmas é grande, causando a destruição destes padrões altamente conservados.

Os nossos resultados demonstraram que Programação Genética é bastante eficaz a gerar datasets sintéticos relativamente a pesquisas aleatórias, apresentando melhorias de 30% na qualidade dos mesmos. Além disso, o nosso método é indiferente ao tamanho do espaço de procura, pois mostrámos que sequências longas, ao nível da resolução do modelo (10.000 nucleótidos), não impactaram a qualidade final dos datasets gerados.

Igualmente importante foi estudar o impacto da gramática usada na qualidade dos dados que são gerados. Os resultados demonstrados previamente usaram uma gramática que codifica perturbações aleatórias à sequência original (p.e. uma inserção de 4 nucleótidos, `ATTT`). No entanto, desenvolvemos também uma gramática mais próxima do domínio, ao codificar as perturbações usando padrões já conhecidos onde reguladores do *splicing* se ligam. Esta gramática restringe ainda mais o espaço de procura, pois as inserções ou deleções na sequência estão sujeitas apenas àquilo que já é conhecido biologicamente. Os resultados mostraram que esta gramática cobre de forma menos eficaz o espaço de previsão do SpliceAI. Isto sugere implicitamente que o modelo pode ter aprendido representações sobre o *splicing* que são desconhecidas aos humanos. Essas representações justificam o melhor desempenho da gramática de perturbações aleatórias, que é menos restrita pelo conhecimento atual.

Por último, integrámos estes conceitos de gramáticas e geração de sequências num software com o objetivo de fazer experiências *in silico* sobre *splicing*, usando os modelos de *deep learning* como fonte de conhecimento para guiar as pesquisas. O software - de seu nome DRESS - é flexível e permite interrogar os modelos a vários níveis. No entanto, importa sublinhar que o software é tão útil quanto os modelos sejam capazes de representar biologia real e permitam a sua decomposição em conceitos compreensíveis ao ser humano.

Em suma, esta tese explorou o uso de modelos estado da arte para a previsão e estudo do *splicing* do RNA. Demonstrámos que o potencial destes modelos, conjugado com técnicas de interpretabilidade ajustadas ao seu domínio de aplicação, pode representar um avanço real na forma de conduzir investigação biomédica. Porém, ainda está por ser desenvolvido um modelo eficaz que permita estudar *splicing* com especificidade ao tipo celular.

Palavras Chave: *Deep Learning*, *Splicing* do RNA, Previsão de Variantes Genéticas, IA Explicável, Programação Genética.

Contents

List of Figures

List of Tables

List of Abbreviations

3'SS Three-prime Splice Site. 8, 11, 12, 27, 78

5'SS Five-prime Splice Site. 8, 11, 12, 27, 78

ACMG-AMP American College of Medical Genetics and Genomics and the Association for Molecular Pathology. 52

AI Artificial Intelligence. 13, 14

ANNs Artificial Neural Networks. 14, 15, 27

auPRC area under the PR curve. 43, 44, 45, 46, 47, 48, 154, 160, 161

auROC area under the ROC. 41, 154

BatchNorm Batch Normalization. 17, 25

bp base-pairs. 1, 7, 39, 62, 68

BP Branch Point. 8, 31, 32, 43, 44, 45, 78, 154

CATs clinically accessible tissues. 53

CNNs Convolutional Neural Networks. 14, 15, 17, 18, 21, 22, 25, 27, 118, 119, 120

DFIM Deep Feature Interaction Maps. 21

DNA Deoxyribonucleic acid. 7

DNN Deep Neural Network. 14, 15, 19, 21, 22, 23, 76, 78, 80

dPSI delta PSI. 24, 79

DRESS Deep learning based Resource for Exploring Splicing Signatures. 105, 114

EAs Evolutionary algorithms. 22, 23

ESEs Exonic Splicing Enhancers. 10, 11

ESSs Exonic Splicing Silencers. 10, 11

FN False Negatives. 37

FP False Positives. 37

GANs Generative adversarial networks. 83

GGGP Grammar-Guided Genetic Programming. 23, 81, 85, 88, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 103, 104, 117, 175, 176

GP Genetic Programming. 23, 81, 83, 84, 85, 90, 99, 118

GPUs Graphical Processing Units. 13

GTEx Genotype-Tissue Expression. 25, 27, 45, 47, 48, 50, 51, 52, 53, 153, 157, 162

HGMD Human Gene Mutation Database. 10, 33, 34, 35

hnRNP heterogeneous nuclear ribonucleoprotein. 8, 10, 66, 70, 71

HPO Human Phenotype Ontology. 51, 52, 157

IAD Increase Archive Diversity. 87, 88, 90, 91, 92

ISEs Intronic Splicing Enhancers. 8, 11

ISM *In silico* mutagenesis. 18, 19, 20, 21

ISSs Intronic Splicing Silencers. 10, 11

LIME Local Interpretable Model-agnostic Explanations. 22, 23

LLMs Large Language Models. 118, 119

MCC Matthews Correlation Coefficient. 154

MLP Multilayer Perceptron. 14

MPRA Massively Parallel Reporter Assay. 27, 33, 34, 44

MPSAa Massively parallel splicing assays. 116

mRNA messenger RNA. 7

NAMs Neural Additive Models. 22

NGS Next Generation Sequencing. 24

NLP Natural Language Processing. 1, 14, 118

NMD Nonsense Mediated Decay. 11, 107

PR Curves Precision-Recall Curves. 154

PSI percent-spliced-in. 24, 57, 89, 96, 107, 110, 112

PSSM Position Specific Scoring Matrix. 47

PTC Premature Termination Codon. 11

PWM Position Weight Matrix. 22, 66, 97, 99, 100, 101

PY Polypirimidine. 8

PyPI Python Package Index. 38

RBNS RNA Bind-N-Seq. 117

RBP RNA-binding protein. 8, 10, 11, 57, 58, 60, 66, 78, 82, 97, 116, 117

ReLU Rectified Linear Unit. 15, 16, 17, 19, 25

RNA Ribonucleic acid. 7

ROC Receiving Operating Characteristic. 42, 154

RS Random Search. 90, 99

SF1 Splicing Factor 1. 8

shRNA Short Hairpin RNA. 58

snRNAs small nuclear RNAs. 7

snRNPs small nuclear Ribonucleoproteins. 7

SNV Single Nucleotide Variant. 18, 44, 65, 83, 86, 93, 94, 116, 151, 153, 178

SR Serine/Arginine. 8, 10

SVMs Support Vector Machines. 76

TP True Positives. 37

TPE Tree-structured Parzen Estimator Approach. 90

U2AF1 U2 Auxiliary Factor 35-kDa subunit. 8

U2AF2 U2 auxiliary factor 65-kDa subunit. 8

UTRs untranslated regions. 36, 120

VAEs Variational Autoencoders. 83

VCF Variant Call Format. 4

VEPs Variant Effect Predictors. 4, 29, 31, 32, 36, 37, 38

VETA Variant prEdiction Tools evAluation. 32

VUS Variant of Uncertain Significance. 12, 41

WES Whole Exome Sequencing. 1, 13, 178

WGS Whole Genome Sequencing. 1, 11, 30, 34, 177

xAI Explainable AI. 4, 5, 18, 22

Glossary

BERT Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based model originally developed for NLP tasks. It was pre-trained to learn deep bidirectional representations by jointly conditioning on both left and right context using multi-head self-attention mechanisms in all layers. This enabled the model to understand the context of a word based on its surroundings. Its modeling objectives included Masked Language Modeling (predict masked tokens based on their context) and Next Sentence Prediction (predict whether sentence B follows sentence A), which lead to a comprehensive understanding of both token-level and sentence-level relationships. It has also been adapted for applications beyond NLP, including modeling DNA sequences. 118, 119

Bi-directional Long Short Term Memory A type of recurrent neural network that processes data in both forward and backward directions. This dual approach allows BiLSTMs to capture context from both past and future information within a sequence. They are composed of two LSTM layers: one processing the sequence from start to end, and another from end to start. This architecture is particularly effective for tasks like sequence labeling or time-series prediction. 27

Central Dogma of Molecular Biology Describes the general transfer of sequence information in living organisms. Each genome is defined by a set of chromosomes that are composed by long DNA molecules. DNA carries genetic information in the cells and its sequence is unique to each individual. The dogma refers that DNA can be converted both in DNA (via DNA replication, essential for cellular division, where two identical DNA copies are produced from the original DNA template) and RNA (via transcription, where DNA is replicated in the form of a new molecule, pre-messenger RNA (pre-mRNA)). This pre-mRNA molecule must be processed to form a mature template for protein synthesis. pre-RNA processing includes 5' capping, introns splicing, cleavage of mRNA at the 3' end and polyadenilation. Mature mRNA are then ready to be translated, in which ribosomes synthesize proteins, which then will be folded and ready to perform their functions in the cell. 7

chromatin Complex of DNA and proteins whose primary function is to package DNA molecules into more compact and denser structures. The protein components of chromatin are called histones, which bind to DNA and serve as anchors around which DNA wraps. There are two main chromatin states traditionally described, which relate to the level of DNA compaction: euchromatin has a less compact structure, which allows DNA to be accessible for active RNA transcription; heterochromatin, on the other hand, is a densely packed state that makes it less accessible, thus being associated with gene silencing and maintenance of chromosome integrity. 10, 14, 115

eCLIP Enhanced CLIP (eCLIP) is a method to capture RNA regions bound by RNA-binding proteins (RBPs). Very briefly, the protocol works by UV-crosslinking RNA-protein complexes, immunoprecipitation of the RBP of interest along with the crosslinked RNA, and conversion of that RNA into high-throughput sequencing libraries for sequencing. Bioinformatics analysis are then performed to identify significantly enriched genomic regions compared to control samples (e.g, samples with no specific antibodies), thereby improving the signal-to-noise ratio and capturing specific RBP binding sites. 24, 72

gene knockdown Gene knockdown is a wet lab experiment to temporarily stop or decrease the expression of a target gene. By reducing the expression of a specific gene, one can learn about its function in the cell. For example, one can analyze gene expression changes after a gene knockdown to get new insights into gene regulatory networks and signalling pathways the gene is involved in. 24, 77, 95

genomics It is a branch of biology focused on the comprehensive study of genomes, including their function, structure, and evolution. This field has advanced significantly due to the "omics" revolution, which introduced technologies that enable detailed quantitative analysis of numerous cellular mechanisms, including splicing, transcription factor binding, and alternative polyadenylation. 13

Gradient Descent An optimization algorithm used to minimize a loss function by iteratively moving in the direction of steepest descent of the function's gradient. At each iteration, a given step (learning rate) is taken in an attempt to decrease the loss function until the algorithm converges to a minimum. In other words, Gradient Descent is implemented by computing the gradient of the loss function with respect to each model parameter and adjusting the parameters accordingly. 15

Position Weight Matrix PWM, a widely used structure for representing sequence motifs in biological sequences. It is constructed based on the frequencies of nucleotides observed at each position in a set of aligned sequences. By using PWMs, one can identify regions of sequences that are evolutionarily conserved and likely to have functional significance. 16

sequencing fragments Sequencing fragments are the smaller pieces of the original RNA molecules created during the library preparation for RNA-Seq experiments. Although the original RNA molecules can be thousands of nucleotides long, they are fragmented into shorter pieces during library preparation. These fragments are then input into sequencing instruments, such as those from Illumina, which typically produce paired reads that are 100-150 base pairs in length. 24

splicing isoforms RNA molecules produced from a single gene that result in different mRNA transcripts through the process of alternative splicing. 9, 24

transcription factor Transcription factors are proteins that regulate gene expression by controlling the rate of transcription from DNA to RNA. They bind to specific regions in the DNA sequence and have the ability to turn on and off genes, so that they are expressed in a coordinated way, in the right cell at the right time. Transcription factors work alone or with other proteins by either promoting or blocking the recruitment of RNA polymerase, the enzyme that transcribes DNA to RNA. 14, 18

transcription elongation Regulated process in which an RNA chain complementary to the template strand of DNA is synthesized as RNA polymerase moves along DNA. 10

transcriptome sequencing Technology that sequences the RNA present in a biological sample at a particular moment in time. While genome sequencing targets the DNA, which is (roughly) the same in all cells, this approach captures the subset of genes that are expressed in a given context, thus holding many applications towards understanding the basis of many molecular processes. 24

Chapter 1

Introduction

Deep learning models are playing a transformative role in society, enabling breakthroughs across multiple domains, including finance [1], Natural Language Processing (NLP) [2], or healthcare [3]. A key attribute of deep learning models is their ability to process raw data and automatically learn the relevant representations for a given task. For example, in the medical field, a deep learning model can analyze raw images from an MRI or CT scan to classify or segment the image into different classes, often surpassing human-level performance [4]. Similarly, in life sciences, deep learning models trained on DNA sequences can effectively predict molecular properties, such as gene activity levels or the likelihood that a genetic mutation will cause disease [5, 6]. In addition, these models can be used to advance our scientific understanding of gene regulation mechanisms, including RNA splicing.

RNA splicing is a process where non-coding regions of genes, called introns, are removed from a pre-mRNA molecule, and exons are joined to form a mature mRNA molecule that can be translated into a protein. Splicing is primarily controlled through ill-defined regulatory signals dispersed across exonic (coding) and intronic (non-coding) sequences and is essential for cellular function. In addition, it is tightly regulated across different tissues and cell types [7]. Genetic mutations disrupting the splicing mechanism are a primary cause of human diseases [8], including cancer [9], making the identification of these mutations in clinical practice crucial for diagnosis and treatment.

Classical Whole Exome Sequencing (WES) and targeted gene panels are widely used genetic testing strategies in clinical settings. However, these approaches typically target coding regions and cover less than 2% of the genome, leaving a significant portion, including intronic regions, unexplored. As a result, more than 50% of patient diagnoses remain unresolved [10, 11]. With the advent of Whole Genome Sequencing (WGS), and the possibility to apply it at the population scale [12, 13], rare intronic variation can be identified at unprecedented levels. However, introns in humans are much larger than exons (median length of 1742 base-pairs (bp) vs. 121bp [14]), with thousands of introns longer than 50Kbp [15]. This size discrepancy

makes handling candidate variants in intronic regions particularly challenging, as the sheer amount of detected variants complicates functional interpretation [16], particularly for variants affecting RNA splicing [17]. Given the importance of intronic regions for splicing regulation, deep learning models trained on sequences spanning entire genes can be used to predict and prioritize splicing-related mutations. However, their performance in deep intronic regions has not been independently and thoroughly evaluated. Unfortunately, assessing their effectiveness requires high-quality labeled variant datasets, which are historically scarce in these regions. In addition, there is a lack of a streamlined framework for independent benchmarking that accommodates the large heterogeneity of the growing number of computational models.

Beyond their predictive capacity, we must consider whether highly performant deep learning models have learned complex underlying representations about the problem domain. For example, if a model accurately predicts a disease-causing splicing variant, it may have learned about the splicing mechanism itself, suggesting that these models can be used to extract biological insights and drive scientific discovery. However, understanding how these models arrive at predictions remains challenging due to their complex, highly-parameterized function, making them difficult to interpret [18]. Despite progress in designing interpretable-by-design deep nets [19–21], post-hoc interpretation techniques are the most common strategy for studying model learning (via input perturbations [5, 22] or attribution-based methods [23–25]). While these methods reveal important patterns within sequences, they do not infer semantic rules of the regulatory landscape without additional analyses [26, 27]. Moreover, integrated analyses using deep learning models as "oracles" - in silico ground truth of experimental assays - are missing in the context of splicing regulation.

An alternative approach is to use inherently interpretable surrogate models to emulate the black-box model. Although global surrogates have been proposed for explaining complex models in other domains [28, 29], applying them in genomics remains challenging due to the intricate combinatorial complexity within regulatory DNA [30]. A more practical strategy is training surrogate models on smaller data subsets to achieve local explanations [31–34]. Local surrogates require generating synthetic sequences to augment the datasets locally by employing perturbations, while avoiding distribution shifts. Ideally, the generated sequences should remain syntactically similar but semantically diverse, enabling the prediction landscape to be covered effectively. In the context of genomics, an example of a synthetic sequence complying with these requirements would be a sequence harboring a single nucleotide change relative to the real sequence (syntactically similar) that leads to a large change in the model prediction (semantically different). However, generating such synthetic sequences through random perturbations cannot guarantee these properties.

Domain-specific properties can further complicate data augmentation. RNA splicing studies use sequences requiring specific spatial properties, such as exon/intron structure awareness. Handling these intervals in genomics can be laborious, often requiring error-prone preprocessing

tasks, such as extracting and tracking splice site locations. Additionally, naive sequence perturbations may not be meaningful (e.g., disrupting a splice site will likely disrupt splicing, not revealing any new information). Therefore, user-friendly software that embeds domain knowledge of splicing for in silico sequence manipulation guided by deep learning oracles is lacking. Such software would enable cost-effective, domain-oriented in silico experiments to generate new scientific hypotheses.

1.1 Objectives

Based on the problems outlined above, this thesis addresses the question of whether sequence-based deep learning models are effective scientific tools for RNA splicing research. To this end, we propose tackling two central challenges:

- **Variant effect prediction**. This task refers to measuring the impact of a genetic variant on a given biological process or function. Accurate variant effect prediction is crucial for diagnosing genetic disorders, developing targeted therapies, and advancing personalized medicine. We aim to test these models on a challenging task, focusing on splicing variants occurring in the often-overlooked intronic regions of the human genome.
- **Model interpretability**: We aim to investigate whether deep learning models have learned underlying biological principles and, if so, provide appropriate tools to interrogate them.

1.2 Methodology and contributions

We now outline the high-level methodology and main contributions of this thesis. Additional details, including software availability, datasets, and efforts for reproducibility, are provided in the respective chapters.

1.2.1 Variant effect prediction

For the first challenge, we undertook an extensive effort to comprehensively benchmark computational models capable of predicting, to some extent, a quantitative measure of the effect of an intronic genetic variant. This task, in particular, motivated work in the following areas:

- Data curation. We manually collected from the literature a dataset of experimentally validated disease-causing deep intronic splicing variants. In addition, when possible, we collected and standardized the different molecular mechanisms by which these variants disrupt splicing, allowing for finer-grained benchmarks. These datasets serve as an independent gold standard for future model testing.

- Model selection. We cataloged a wide variety of models to be benchmarked, including not only deep learning models but also classical variant pathogenicity predictors.
- Data preprocessing and engineering. Given that different models target different questions and require different inputs (e.g., a model that predicts splice sites is different from a model that directly predicts variant effects), we developed a utility to effectively generate the proper input for a subset of the models, and process their output into standard Variant Call Format (VCF) format.
- Model benchmarking. We developed a software called VETA that automates the benchmarking of Variant Effect Predictors (VEPs). Most benchmarking results presented in the thesis were obtained using this software.

Accordingly, work related to this challenge gave rise to the following peer-reviewed publications:

L. Lopes, **P. Barbosa**, M. Torrado, et al., "Cryptic Splice-Altering Variants in MYBPC3 Are a Prevalent Cause of Hypertrophic Cardiomyopathy", *Circulation: Genomic and Precision Medicine*, 2020 [35].

P. Barbosa, M. Ribeiro, M. Carmo-Fonseca, A. Fonseca, "Clinical significance of genetic variation in hypertrophic cardiomyopathy: comparison of computational tools to prioritize missense variants", *Frontiers in Cardiovascular Medicine*, 2022 [36].

P. Barbosa, R. Savisaar, M. Carmo-Fonseca, A. Fonseca, "Computational prediction of human deep intronic variation", *GigaScience*, 2023 [37].

1.2.2 Model interpretability

To address model interpretability, we focused on SpliceAI [38], a state-of-the-art widely used deep learning model that predicts the probability that any given position in the sequence would be used as a splice site. In particular, we investigated whether SpliceAI has learned meaningful knowledge about alternative splicing and proposed a novel paradigm to study deep learning models in genomics, via evolutionary algorithms constrained with grammars. This challenge led to the following research topics:

- Studying the model via domain-aware ablations. We leveraged publicly available RNA-Seq data to generate datasets for ablation studies. These studies aimed to understand the model's sensitivity to binding motifs of RNA-binding proteins. Along the way, we designed a pipeline to automate the analyses, focusing on creating outputs prepared for downstream interpretability queries.
- Synthetic data generation for local Explainable AI (xAI). To address the challenge of generating synthetic sequences for local surrogate models, we propose a novel approach that combines genetic algorithms with domain-aware constraints via grammars. We designed two distinct grammars that encode domain knowledge of splicing, guiding the generation

of meaningful synthetic sequences more efficiently.

- Streamlining model interrogations. Taking advantage of the grammar guided framework for synthetic data generation, we developed a software project called DRESS, which enables flexible *in silico* splicing interrogations of deep learning models. We demonstrated its capabilities by interrogating Pangolin, another sequence-based model, on sequences that are tightly regulated during cardiac development.

The work proposing synthetic data generation resulted in a paper published at GECCO'24, as well as my involvement in the evaluation of its underlying Genetic Programming framework:

G. Espada, L. Ingelse, P. Canelas, **P. Barbosa**, A. Fonseca, "Data Types as a More Ergonomic Frontend for Grammar-Guided Genetic Programming", *GPCE 2022*, 2022 [39].
P. Barbosa, R. Savisaar, A. Fonseca, "Semantically Rich Local Dataset Generation for Explainable AI in Genomics", *GECCO' 24*, 2024 [40].

Other publications not directly associated with the main contributions of this thesis are listed in Appendix D.

1.3 Structure

Apart from this introductory chapter, the document is organized as follows:

- Chapter 2 (RNA Splicing background) provides a background on RNA splicing, the domain of application of this thesis.
- Chapter 3 (Deep learning in genomics research) discusses deep learning models and how to interpret them, particularly in the context of genomics and RNA splicing.
- Chapter 4 (Predicting intronic variants affecting the splicing mechanism) presents an independent benchmark assessing the capacity of computational approaches to predict deep intronic variants.
- Chapter 5 (Interpreting SpliceAI) contains an in-depth analysis of SpliceAI, employing large-scale ablation studies on tailored datasets built for this task.
- Chapter 6 (Semantically-rich synthetic dataset generation with constrained Genetic Programming) proposes a grammar-guided Genetic Programming approach to generate semantically rich synthetic local datasets for xAI.
- Chapter 7 (DRESS: a flexible framework for splicing interrogations guided by deep learning models) presents a toolkit for flexible *in silico* splicing interrogations using deep learning models as oracles.
- Chapter 8 (Discussion and conclusion) closes the thesis by providing an overall discussion and considerations on future research directions.

Chapter 2

RNA Splicing background

This chapter delves into the domain of RNA splicing, the primary application area of this thesis. We begin by providing a biological background on the splicing mechanism and its strong association with disease.

2.1 Mechanisms of RNA Splicing

The genome of a species contains the complete set of instructions for an organism's development, starting from the first cell. It is the most fundamental source of information each individual carries. The human genome, for instance, is approximately 3 billion bp long, with each cell containing its Deoxyribonucleic acid (DNA) densely packed within the nucleus. Two of the main cellular building blocks, RNA and protein molecules, are synthesized from DNA through gene expression, which involves several steps (see Central Dogma of Molecular Biology). One crucial step in this process is RNA transcription, where the DNA sequence is transcribed into precursor Ribonucleic acid (RNA), known as pre-messenger RNA (mRNA). This precursor is then processed to produce mature mRNA. A key stage in this processing is RNA splicing, which is the focus of this work.

Splicing is an mRNA processing event that takes place in the nucleus of cells. It involves the removal of introns from the pre-mRNA and is carried out by a complex cellular machinery known as the spliceosome. This process occurs co-transcriptionally, meaning it takes place during transcription elongation.

The spliceosome is a large protein-RNA complex consisting of five small nuclear RNAs (snRNAs) (U1, U2, U4, U5, U6) and approximately 200 proteins [44]. Some of these proteins associate directly with snRNAs to form small nuclear Ribonucleoproteins (snRNPs), while others, non-snRNPs, bind directly to the pre-mRNA. The spliceosome is assembled in a stepwise fashion on every newly synthesized intron, with its components recruited through base-pairing interactions between the spliceosomal snRNAs and conserved sequences in the pre-mRNA

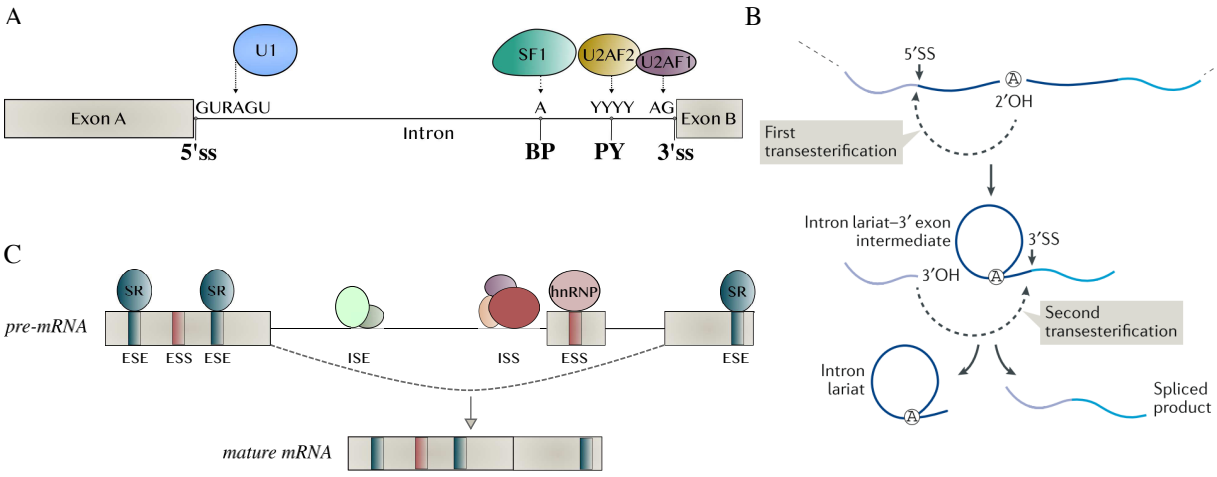

Figure 2.1: Elements involved in pre-mRNA splicing of an intron. **A** - Canonical pre-mRNA elements are recognized to trigger the assembly of the spliceosome. These elements include the Five-prime Splice Site (5'SS) at the beginning of the intron, the Three-prime Splice Site (3'SS) at the end, the Branch Point (BP) sequence located upstream of the 3'SS, and the Polypirimidine (PY) tract between the branch point signal and the 3'SS. Apart from the highly conserved GU and AG dinucleotides at the first and last two positions of introns, respectively, splicing signals in the mRNA sequence are degenerate, meaning that various sequences and base-pairing schemes can assemble the spliceosome. Pre-spliceosome assembly involves the binding of U1 snRNP to the 5'SS and subsequent recognition of BP, PY, and 3'SS by non-snRNP splicing factors, Splicing Factor 1 (SF1), U2 Auxiliary Factor 35-kDa subunit (U2AF1), and U2 auxiliary factor 65-kDa subunit (U2AF2). This stage, referred to as Complex E, is essential for the subsequent recruitment of additional snRNPs and downstream assembly of a catalytic spliceosome (not shown). **B** - Two transesterification reactions that biochemically define splicing (adapted from [41]). In the first reaction, the 2'-hydroxyl group of the BP adenosine attacks the phosphate group at the 5'SS, generating a free 5' exon and an intron lariat-3' exon intermediate. Consequently, the exposed 5' exon 3'-hydroxyl group attacks the 3'SS, cleaving the lariat-structured intron and ligating the two exons. The exons are released from the spliceosome, and the intron lariat is eventually degraded. **C** - Splicing regulatory sequence elements. These are named based on their location and function: Exonic Splicing Enhancer (ESE), Exonic Splicing Silencer (ESS), Intronic Splicing Enhancers (ISEs), Intronic Splicing Silencer (ISS). Serine/Arginine (SR) proteins usually bind to ESEs and promote exon inclusion [42]. Conversely, the classical role of the heterogeneous nuclear ribonucleoprotein (hnRNP) family is to repress splicing when binding to ESSs, although the picture is less clear when these proteins bind to intronic elements [43]. In this example, SR proteins are bound to enhancers in the left and right exons, while an hnRNP binds to the middle, hindering its inclusion in the mature mRNA.

(*cis*-acting elements, Figure 2.1A). During splicing, two transesterification reactions occur, excising the intron and ligating the exons (Figure 2.1B). Additionally, recognition of splice sites by spliceosome components can be enhanced or repressed by additional proteins, named RNA-binding protein (RBP)s (*trans*-acting elements), which bind to splicing regulatory elements in exons and introns (Figure 2.1C) [43]. The combinatorial usage of all these elements ultimately defines the final structure of the spliced transcript. The next section explores this topic in detail.

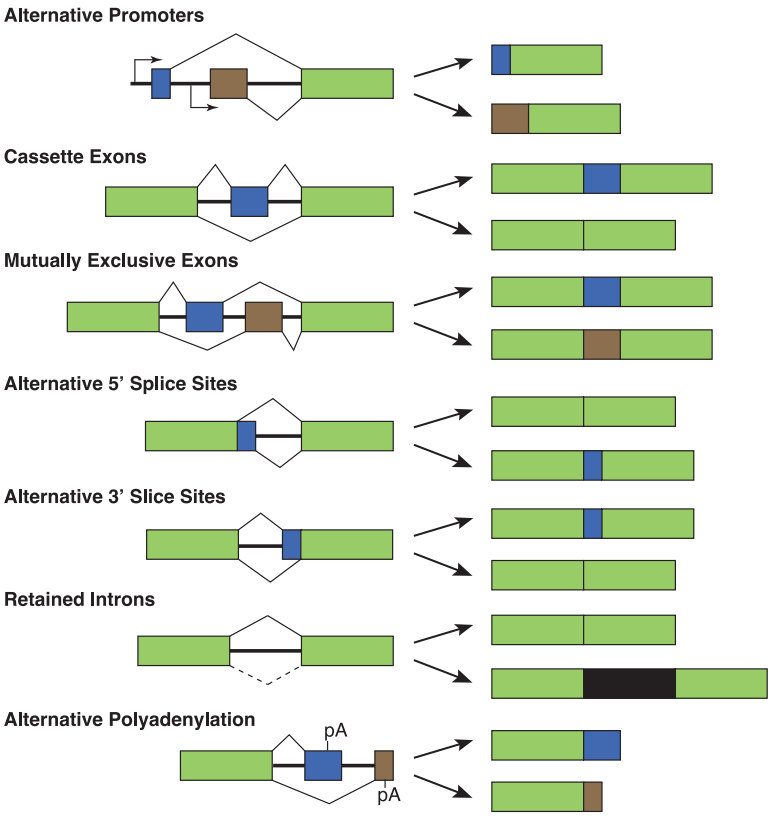

Figure 2.2: Mechanisms of alternative splicing (extracted from [45]). Each mechanism generates different mature RNAs. Exons are represented as boxes, introns as lines. While constitutive exons are displayed in green, alternatively spliced sequences are presented in blue or brown. When an intron is not spliced out it results in intron retention, depicted by a black box in the final transcript.

2.1.1 Alternative Splicing

Alternative splicing is a mechanism that allows a single gene to generate multiple distinct mRNAs, known as splicing isoforms, by assembling different combinations of exonic segments from the same template DNA. This mechanism expands the proteome diversity encoded by the genome. There are several types of alternative splicing (Figure 2.2), with exon skipping being the most common. In exon skipping, a single exon can be included or excluded from the mature mRNA, resulting in *cassette exons*. Conversely, exons that are always included are referred to as *constitutive exons*. Alternative splicing is regulated in different cellular environments, and this regulation is tissue-specific. For example, sequences are spliced differently in brain cells compared to muscle cells. Additionally, regulated shifts in splicing patterns have been shown to be coordinated during development and cell differentiation, underscoring the critical role of splicing networks in tissue function and identity [7].

The mechanisms regulating splice-site selection and, consequently, alternative splicing are

complex and partially remain poorly understood. However, certain key players are known to be involved in these decisions. RBPs interact with pre-mRNA and modulate how efficiently the spliceosome components recognize the splice sites. The most studied classes of RBPs are members of the SR and hnRNP families. SR proteins generally activate splicing, promoting exon inclusion by binding to Exonic Splicing Enhancers (ESEs) and interacting with snRNPs to strengthen exon definition. Conversely, members of the hnRNP family typically repress splicing by binding to silencer sequences (Exonic Splicing Silencers (ESSs), Intronic Splicing Silencers (ISSs)) and interfering with the core splicing machinery's ability to engage splice sites [46]. Nevertheless, these roles are not absolute. There is plenty of evidence showing that these proteins can have opposite functions depending on their binding position and contextual neighborhood [43, 46, 47].

Many RBPs are ubiquitously expressed across tissues, while only a few regulate specific tissues [7]. How can such a limited number of regulators control the vast array of alternative splicing events in human tissues? This diversity is explained by other factors, such as RNA structure. mRNA structures are critical in determining the accessibility of splice sites and their regulatory sequences. Several proteins regulate splicing by modulating RNA structures or binding to structural elements [48], thereby enhancing or repressing splicing.

Additionally, splicing is a highly dynamic process involving numerous kinetic steps. The timing of the splicing reaction relative to transcription elongation is influenced by several factors, including gene architecture features such as exon or intron size, transcription rate, and consecutive splice site competition. These factors define the accessibility of spliceosome components to core splicing motifs [49]. Specific chromatin modifications [50] and chemical RNA modifications [51] are also associated with splicing modulation, adding further layers of regulation to alternative splicing networks.

2.1.2 Splicing and disease

Given splicing complexity, it is not surprising that this RNA processing step is vulnerable to both hereditary and somatic genetic variants implicated in disease. It is estimated that 10 to 50% of all monogenic disease-causing variants affect pre-mRNA splicing [38, 52, 53]. Additionally, cancer driver mutations are often associated with splicing alterations, particularly in variants that occur in genes encoding core components of the splicing machinery [41].

There have been continuous efforts to systematically catalog disease-causing variation in databases such as ClinVar [54] or the Human Gene Mutation Database (HGMD) [55]. These resources show the enrichment of splicing-related variants in the vicinity of splice site regions. Partly, this reflects a biological reality, where the sequence around the splice sites is particularly dense in splicing-relevant information. However, this enrichment may also stem from the easier detection of splice site mutations, as well as biases in clinical guidelines for variant interpretation

that may contribute to underestimating the significance of deep intronic splicing mutations, as there is a lack of standardized criteria for their interpretation [56, 57].

Mechanisms

Of the *cis* variants that affect splicing, those that disrupt splice sites (typically AG for the 3'SS and GT for the 5'SS) or the consensus region around them (nucleotides -12/+2 around the 3'SS and -3/+6 around the 5'SS) have been studied the most thoroughly. Variants in these regions, especially if they affect the splice sites themselves, are fairly easy to recognize because the sequences are short and adhere to a highly conserved motif [8]. In contrast, other splicing variants can impact the binding of regulatory factors to splicing enhancers or silencers (ESEs, ISEs, ISSs and ESSs) [46]. Because they are poorly defined sequence motifs which occur throughout the gene, it is difficult to identify them - and even more difficult to know when a mutation has disrupted them.

Disruption of splicing information encoded by variants in *cis* elements can lead to aberrant splice events such as exon skipping, full intron retention, or exon shortening or lengthening. Splicing variants can also create entirely new exons, known as *pseudoexons*. This can occur when a mutation generates a novel splice site or when an existing but inactive (*cryptic*) splice site is activated by the creation of an enhancer motif or the disruption of a silencer motif [58]. Mutations in genes that regulate splicing (*trans* elements), such as core spliceosome genes and RBPs, may also contribute to disease phenotypes. These mutations often have larger effect sizes because they typically regulate the splicing of multiple genes [41].

Deep intronic variants

Given the short DNA/RNA alphabet, numerous cryptic splice sites within extensive intron regions resemble authentic splice sites. These are usually repressed because their location do not favor interactions with the splicing machinery, and they are flanked by high densities of splicing silencer motifs [58]. This repression is crucial for maintaining transcriptome integrity, ensuring that mRNAs remain in frame to code for functional proteins [46].

Mutations deep within introns can disrupt this balance, activating pseudoexons and producing aberrant transcripts. This often leads to the introduction of Premature Termination Codon (PTC) in the mRNA, targeting the transcript for degradation by Nonsense Mediated Decay (NMD) and resulting in the loss of the protein product (Figure 2.3). Although previously overlooked, deep intronic mutations are now recognized as a significant cause of human disease [59, 60].

However, their clinical interpretation remains challenging. Introns are large, making the search space extensive, and mutations with no effect greatly outnumber rare splice-affecting variants. As a result, deep intronic variants detected through WGS often end up labeled as

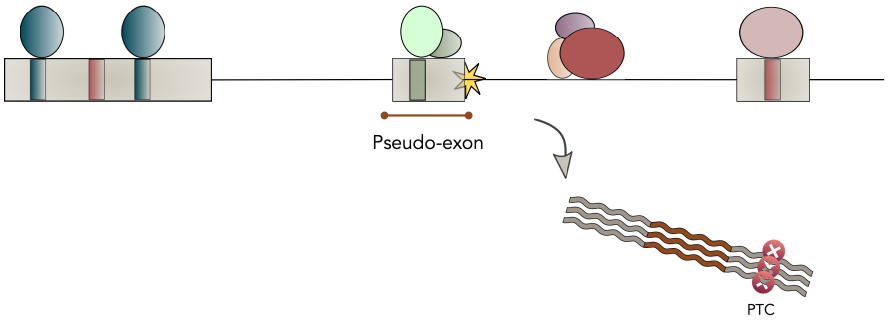

Figure 2.3: Deep intronic variant triggering pseudo-exon inclusion. A genetic variant (yellow asterisk) creates a splice donor (5'SS) in the middle of an intron. An upstream cryptic splice acceptor (3'SS) is activated leading to pseudo-exon inclusion (red sequence stretches in the lower part of the figure), and subsequent degradation by NMD.

Variant of Uncertain Significance (VUS) [61]. Computational tools can help prioritize variant candidates, but their efficacy in predicting deep intronic splicing effects remains to be addressed.

Chapter 3

Deep learning in genomics research

This chapter introduces key concepts essential for understanding the themes of this thesis. This includes an introduction to deep neural networks, appropriate architectures to model biological sequences and a discussion of strategies for studying and interpreting deep neural nets in genomics. Finally, we offer a historical perspective on the use of deep learning strategies modeling RNA splicing. While this chapter serves as the background for the work, we also highlight an early contribution that aligns with the concepts presented:

- We demonstrated the utility of deep learning models to score disease-causing variants in intronic regions often missed by WES studies. I conducted the computational analysis for this study.

  L. Lopes, **P. Barbosa**, M. Torrado, et al., "Cryptic Splice-Altering Variants in MYBPC3 Are a Prevalent Cause of Hypertrophic Cardiomyopathy", *Circulation: Genomic and Precision Medicine*, 2020 [35].

3.1 Deep learning revolution

The rise of deep learning methodologies in the early 2010s has profoundly transformed numerous fields, driving unprecedented progress in technology and science. The emergence of large, high-quality, publicly available labeled datasets, advances in hardware with Graphical Processing Units (GPUs) for training neural networks, and continuous breakthroughs in architecture design have been key factors contributing to the deep learning revolution [62]. In addition, the appearance of powerful libraries such as TensorFlow [63] and PyTorch [64] has democratized access to deep learning, enabling a wide range of research communities to tackle diverse problems. In fact, deep learning has become so central to Machine Learning that when people talk about Artificial Intelligence (AI), they are often referring to models that are themselves deep learning-based solutions.

Specific model architectures have driven advancements in different domains. For example, deep Convolutional Neural Networks (CNNs) [65] are especially well-suited for Computer Vision tasks such as image classification, object detection, and segmentation [66], while Transformers [67] have revolutionized NLP problems like text summarization, question answering, and machine translation [68]. Although primarily optimized for these data modalities, these and other techniques have been successfully adapted to other objectives, including scientific discovery [69]. AI methods can improve the traditional research workflow by streamlining procedures for data collection and processing [70], exploring vast spaces of candidate solutions [71], and providing insights that would otherwise be difficult to obtain [72].

In life sciences, deep learning has been successfully applied to predict molecular phenotypes from genomics data. These models take a DNA sequence as input and aim to predict some property of the sequence's activity, such as gene expression [6], chromatin accessibility [73] or transcription factor binding [26]. The high performance of such *sequence-to-activity* models, combined with the application of explainable AI techniques, holds unprecedented potential to deepen our understanding of cell biology and disease mechanisms.

In the following sections, we introduce Artificial Neural Networks (ANNs) as the foundational concept for deep learning, describe the most widely used neural network architecture for modeling genomics data (CNNs), and discuss techniques for interpreting sequence-based genomics models. Finally, we delve deeper into the domain of application of this thesis: RNA splicing.

3.2 Artificial Neural Networks

ANNs are computing systems inspired by the biological neural networks that compose animal brains. To understand how ANNs are conceptually organized, we can look at the simplest type of networks, the Multilayer Perceptron (MLP). An MLP is a neural network composed of three types of layers: an *input layer* that receives the raw data, one or more *hidden layers*, which perform non-linear transformations of the inputs and a final *output layer*, that returns an output. Each layer is made up of nodes (*neurons* or *units*), each of them fully connected to the neurons of the next layer (each connection transmits a trainable *weight*). When the neural network includes a deep stack of hidden layers, it is called a Deep Neural Network (DNN), or a deep learning model. Typically, an extra *bias* feature is added (bias neuron), which provides every node with a trainable constant value (in addition to the normal inputs/weights that a node receives). An MLP can be viewed as a generalization of (generalized) linear models, meaning that each node in the hidden layers take the sum of their weighted inputs. The key difference is that an activation function σ is applied to this value to introduce non-linearity into the model. The parameterized function of a single MLP layer can be formulated as following:

$$f(x, \theta) = \sigma(x\theta + b) \tag{3.1}$$

where x represents the matrix of input features, θ represents the weight matrix (the parameter values for all the connections, except those from the bias neuron), the b stands for the bias vector that contains all the connection weights between the bias neuron and the hidden nodes, and σ is the activation function. A frequently used activation function is the Rectified Linear Unit (ReLU) defined as $\sigma(z) = \max(0, z)$.

3.2.1 Training deep neural networks

Training neural networks involves adjusting the weights to minimize the loss function, often using optimization algorithms derived from Gradient Descent. While methods like Stochastic Gradient Descent are common, optimizers such as ADAM and RMSprop have gained popularity for their adaptive learning rates and incorporation of momentum-like behavior, which helps on escaping plateaus where the derivative is close to zero [74]. Because computing the gradient of the loss for each parameter would be intractable for DNNs, the *backpropagation* algorithm efficiently computes gradients across the network in just a *forward* and *reverse* pass [75]. Briefly, backpropagation employs the chain rule of calculus to recursively calculate the gradient of the loss function with respect to each parameter. It propagates errors backward through the network, adjusting the weights using the chosen optimizer. This iterative process operates on small subsets of the training data, called *batches*, and completes an *epoch* when all batches comprising the training data have passed through the network once. Training ends when a predetermined number of epochs is reached, or when the model converges to a satisfactory level. At this stage, the trained weights are used to evaluate the model's performance on separate test data, ensuring an unbiased assessment of its generalization power.

3.3 Convolutional Neural Networks

CNNs are a type of ANNs that are composed by specific layers, called *convolutional* layers, that are specialized in the detection of patterns in the data. Neurons in convolution layers are not fully connected to the input data. Rather, each neuron connects to regions of the input space that belong to its *receptive field*. The receptive field is defined by the *filter* dimensions of the layer. As an example, in image classification, a neuron in a convolutional layer connects only to pixels of the image defined on its respective field.

A filter, which can be a small matrix of values, slides the input data and at each location applies linear transformations so that an output value for each location is obtained. The output of this sliding operation is a *feature map*, which highlights the areas of the input data that activate the filter the most. The great thing is that filters do not have to be defined manually:

during training, the convolutional layer will automatically learn the most useful filters for the task - filters are the weights in a convolutional layer. All neurons within a feature map share the same weights, which dramatically reduces the number of parameters of the model. In practice, a convolutional layer usually has multiple filters and outputs one feature map per filter, making it capable of detecting multiple features anywhere in its inputs. This type of layer thrives in array data (e.g., images, videos) because groups of values are often highly correlated, forming distinctive local motifs that can be easily detected. The output of this layer is followed by a nonlinear activation function (e.g., ReLU), similar to fully-connected layers.

In genomics, a 1-dimensional convolutional layer is applied to one-hot-encoded DNA sequences. The filters applied to the sequence are analogous to a Position Weight Matrix scanning the sequence. Importantly, since the filter scanning partially contributes to *translation invariance* - the ability to recognize patterns regardless of their position -, the model can generalize to motif positions not seen during training.

A typical CNN architecture stacks multiple convolutional layers (each followed by non-linear activation functions), interspersed with pooling layers (Figure 3.1). As the network gets deeper, it learns how to combine low-level feature maps into higher-level feature representations. At the top of the stack, fully-connected layers are often added at the end to translate these learned features into predictions for classification tasks.

3.3.1 Pooling

Convolutional layers are often accompanied by *pooling* layers. A pooling layer aims to subsample the input data, reducing computational load, memory usage and number of parameters. Just like a convolutional layer, a pooling layer scans the input data (in this case, the feature maps passed by the previous layer) and aggregates the inputs using an aggregation function such as the *max* or *mean*, returning just a single value to the next layer, which means that it does not have trainable weights. This layer also contributes for translation invariance as the down-sampling of feature maps can make the output more robust (invariant) to changes in the position of the features.

3.3.2 Dilated convolutions

Dilated convolutions are a variant of the standard convolutional layer that allow increasing the receptive field of the network without increasing the number of parameters [77]. This is achieved by using special filters that skip some input values. The dilation rate defines the number of positions to skip, and it typically increases with every subsequent dilated convolutional layer. Dilated convolutions are particularly useful tasks where capturing long-range dependencies is important.

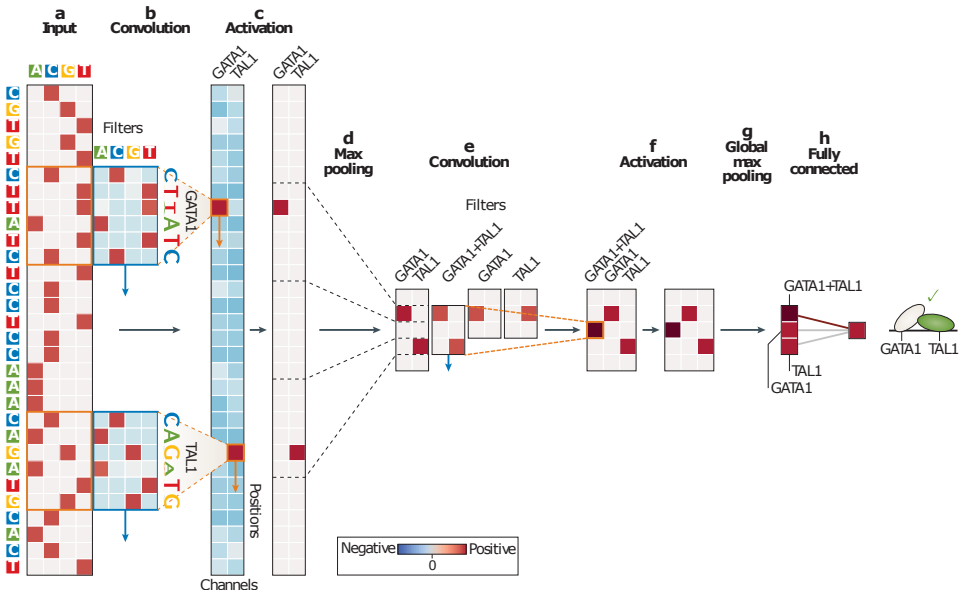

Figure 3.1: Convolutional neural network modelling genomics sequences. **a** - One-hot encoding representation of the DNA sequence. **b** - Filters of the first convolutional layer scan the input sequence. **c** - Negative values are truncated to 0 using the ReLU activation function. **d** - Max pooling operation summarizes contiguous bins of the activation map by taking the maximum value for each channel in each bin. **e** - The second convolutional layer scans the sequence for pairs of motifs and for instances of individual motifs. **f** - Similar to the first convolution, ReLU activation function is applied. **g** - The maximum value across all positions for each channel is selected. **h** - Fully connected layer is used to make the final prediction. Figure and legend adapted from [76].

3.3.3 Regularization

Regularization techniques are used to prevent overfitting and improve convergence during the optimization process by keeping the weights and activations within a reasonable range. Regularization is particularly important to prevent vanishing or exploding gradients, specially in networks with many layers. Common regularization techniques include *Dropout* [78] and *Batch Normalization (BatchNorm)* [79]. In Dropout, a certain percentage of neurons are randomly set to zero during training, forcing the network to not rely too much on single neurons and thus learn more robust feature representations. Batch Normalization normalizes the layer activation by subtracting its mean and dividing by standard deviation across multiple samples in the batch, which results in more stable training as the magnitude of the activations throughout the network is relatively uniform.

3.4 Model interpretation

Thanks to the development of multiple genome-wide assays that generate large collections of sequence data with a given readout, a plethora CNNs have been successfully applied to

predict molecular phenotypes from genomics data. By understanding the properties of the data underlying successful predictions, it is hypothesized that new biological insights can be derived using these models as interrogation tools. We next discuss several *post-hoc* model interpretation techniques that are part of the rapidly evolving field of xAI.

3.4.1 Model-based interpretation

Model-based techniques examine individual components of the model. In genomics, because the filters learned by the first convolution layer may represent short subsequences of binding motifs (Section 3.3), one can apply a softmax transformation (with further scaling) to a convolution weight matrix to visualize the matrix associated with a filter [30]. A more faithful strategy in practice is to search for subsequences in a dataset that activate a given filter the most [80, 81]. Of note, the extraction of these subsequences does not imply that they are important. Additional experiments, such as ablating or masking the filter, are required to measure their impact on the prediction.

3.4.2 Propagation of influence

This class of algorithms operates directly on input sequences by propagating perturbed data through the model and measuring the effect on predictions. They can be grouped into *forward* and *backward* propagation methods.

In silico mutagenesis

Forward propagation methods include *In silico* mutagenesis (ISM), which employs systematic mutations to each nucleotide in an input sequence and computes the change in the model outcome due to each mutation (Figure 3.2A). In contrast to other approaches that are discussed next, ISM truly represents the model's response to genetic mutations at individual positions, making it the standard approach to predict the clinical impact of Single Nucleotide Variant (SNV)s [5]. Although conceptually attractive, ISM is computationally expensive because it involves a forward propagation pass for every mutation tested ($3L$ where L is the length of the sequence). To address this problem, one can limit the analysis to a subset of sequences based on hypothetical insight value (e.g., high score prediction), or restrict the analysis to a subset of sequence regions deemed important for the task. For example, perturbing known transcription factor motifs can hold great explainability value, although it requires a priori knowledge of the motif locations. Recently, a new strategy called fastISM [22] has been developed to accelerate the computation of ISM scores. This method leverages the unique properties of CNNs to avoid redundant calculations and efficiently compute values within the receptive field of a given convolution filter.

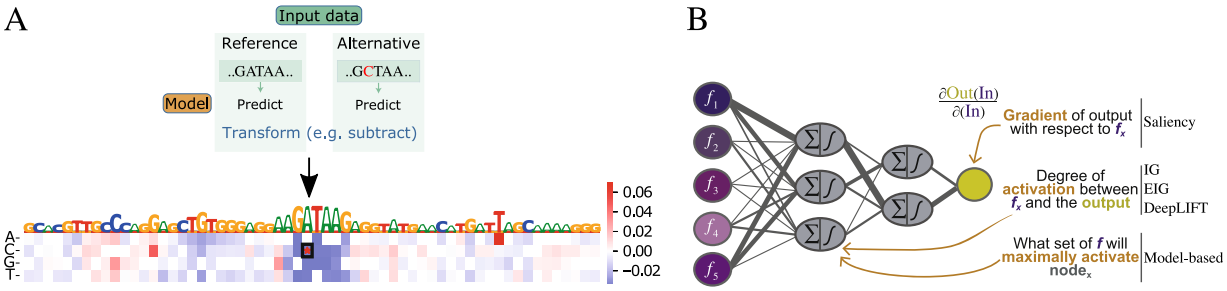

Figure 3.2: Strategies for neural network interpretation. **A** - Example of using ISM to interpret a sequence-based DNN. At the top, model predictions for two sequences are obtained, one for the reference sequence, other with a mutated feature. Feature importance scores are computed by subtracting the scores (or by applying any other transformation). These scores can be visualized as an *attribution map*, in the bottom: each square in the heatmap represents the effect of a single perturbation in the model prediction. Blue and red colors represent a negative and positive effect, respectively. The highlighted square at the center represents the mutated feature in the sequence represented in the top. Image adapted from [82]. **B** - Gradient and model-based approaches for neural network interpretation. IG = Integrated Gradients; EIG = Enhanced Integrated Gradients. Image adapted from [83].

Saliency maps and derivatives

Backward propagation methods address the computational overhead of ISM by using backpropagation to efficiently decompose the output prediction of a model into character-level attribution scores in a single pass [30]. The most straightforward method, commonly referred to as a saliency map ([84]), computes the gradient of the output with respect to the input data (Figure 3.2B). Feature importance correlates with the amount of the gradient they receive after backward propagation. Gradient-times-input [85] slightly extends Saliency maps by doing an element wise multiplication of each input feature with the gradients they receive. In the context of genomics, where inputs are one-hot encoded, this translates to computing the gradient on the bases that are present in the sequence. These approaches can be problematic because activation functions such as ReLU have a gradient of zero when they are not firing. Due to this, Saliency maps fail to highlight inputs that contribute negatively to the output. Small variations to this approach like Deconvolution Networks [86] or Guided Backpropagation [87] have been proposed, and basically differ in the specific backpropagation logic for the activation function.

A major drawback of these methods is the so-called *saturation* problem, which underestimates the importance of features that have saturated their contribution to the output. For example, in a sequence with multiple copies of the same binding motif, the model's overall sensitivity to this motif is spread across the multiple copies. Consequently, this redundancy can cause the individual feature attributions to be underestimated.

Reference-based methods

Reference-based methods were proposed to address the saturation problem. Here, feature attribution reflects the comparison of a neuron's activation to a given input *reference* or *baseline* (Figure 3.2B). This reference is the activation that the neuron would have when given a reference input. For genomics, a mononucleotide or dinucleotide shuffling of the input sequence is considered a reasonable reference [30].

Integrated Gradients [24] proposes an axiomatic approach for feature attribution. It is computed by integrating the gradients along a linear path from the reference input to the actual input, which is approximated by summing the gradients at points along this path. A feature has a high attribution value when the integrated gradient shows it significantly affects the change in activation from the reference to the input. Enhanced Integrated Gradients [25] extends Integrated Gradients by introducing alternative non-linear path formulations.

DeepLIFT [88] differs from Integrated Gradients by comparing the differences in activation between the input and the reference at every node of the network. These differences are propagated backwards through the network using specific rules to ensure that each node's contribution at specific layers is accurately accounted for. These rules address gradient discontinuities at certain layers, with one of the rules shown to approximate Shapley values [89].

Importantly, backpropagation-based methods can be computationally expensive when applied to large multi-task models, such as those predicting multiple molecular phenotypes [80] or returning vector outputs representing quantitative regulatory profiles [26]. In these instances, a separate backpropagation pass is required for each task or output dimension to estimate comparable feature attribution per task. In contrast, ISM might be more efficient, as perturbing individual nucleotides reveals the impact on each output in a single pass.

From local to global interpretations

The propagation methods discussed so far provide local explanations, meaning that they explain a single input sequence. To aggregate results across multiple inputs into a global understanding of feature importance, TFMoDisco [90] was specifically this task. It uses attribution scores from methods like DeepLIFT to extract sequence regions with high attribution scores (*seqlets*), calculates their similarities, and performs clustering based on the similarity matrix between pairs of seqlets. Final consensus motifs are generated upon cluster refinement. Nevertheless, the method is highly dependent on the quality of the attribution scores (e.g., the choice of the reference has a significant impact on the results [88]), and typical challenges of clustering remain, like the choice of the number of cluster and similarity metrics. In addition, TFMoDisco itself does not reveal effects of motif combinations and spacing, requiring additional analysis to understand such motif rules [26].

3.4.3 Probing feature interactions

Current feature attribution methods do not explicitly quantify interactions between features, such as epistatic interactions (e.g., motif cooperation or competition). However, the power of CNNs lies in their hierarchical structure, which allows them to learn complex non-linear interactions between features. Hence, one can naively examine sequences that activate deeper layers (as done in Section 3.4.1 for the first convolutional layer).

Alternatively, tweaked versions of ISM can be applied to experimentally test inserting two motifs into a random sequence (a type of *marginalization* experiment) and observing the effect of perturbing one motif while keeping the other fixed. This setup probes both additive and non-additive effects between motif pairs and can also be used to assess the impact of motif spacing [30]. A related strategy, termed Deep Feature Interaction Maps (DFIM) [91], quantifies similar interactions but instead uses backpropagated-based methods to compute changes in the importance of a target feature when a source feature is perturbed. Recently, an *in silico* perturbation toolkit named CREME [92] was developed to test the contribution of various components on model prediction. These include not only higher-order interaction tests, but also the impact of surrounding context, the sufficiency of individual regulatory elements, and their distance effects.

3.4.4 Transparent models

Unlike *post hoc* methods, which provide explanations for already trained models, training interpretable models involves designing them with transparency at their core. We categorize these models into two types: DNNs with inherently interpretable units (*interpretable-by-design models*) and *surrogate models*, which are simpler, transparent models trained to approximate the behavior of complex black-box models. In addition, we introduce evolutionary algorithms as a potential strategy addressing model interpretability, as they will be relevant for the work presented in this thesis.

Interpretable-by-design models

Interpretable-by-design models embed prior knowledge into the architecture design, ensuring the encoded knowledge is at a level of abstraction that humans can understand. For example, graph-based architectures embedded with ontologies learned to prioritize domain-specific hierarchies and interactions between concepts, such as gene mutants affecting specific gene ontology processes [93] or revealing therapeutic targets for prostate cancer [94]. However, these models are not trained with genomic sequences, which is the focus of this thesis.

Training sequence-based models with interpretable units requires initializing convolutional filters with known motifs [95, 96] or fixing filter lengths to specific problem domains [97].

Importantly, the learned filters may drift away from those used for initialization [30], and applying such constraints may not be feasible for domains with insufficient prior knowledge. Recently, an approach based on Neural Additive Models (NAMs) [98] was proposed for genomics data. Briefly, ExplainNN [21] computes predictions as a linear combination of multiple independent CNNs, each containing a single convolutional layer with a single filter, an exponential activation and two fully connected layers. The learned weights of each unit in the fully connected output layer are interpretable, similar to those in a linear model. However, this linearity can compromise accuracy on complex, realistic genomics tasks. The authors observed that it hampered the model's ability to learn complex interactions between motifs of different transcription factors [21]. Additionally, further analyses are required to convert the filter of each unit into a biologically meaningful Position Weight Matrix (PWM).

Surrogate models

Surrogate models provide an interpretable approximation of complex deep learning models. These simpler models, such as a decision tree or linear regression, are trained to mimic the behavior of the more complex model, offering insights into its decision-making process. Due to the complex and highly parameterized nature of DNNs, surrogate models are often trained to approximate the DNN locally, aiming to explain a single or a few related instances.

The most popular of such surrogate modelling approach is Local Interpretable Model-agnostic Explanations (LIME) [31], which samples data around a target instance and trains a linear model on this synthetic dataset to approximate the predictions of the original model. LIME is limited by the difficulty of defining a meaningful neighborhood when sampling data [99], especially since two close data points may result in very different explanations [100]. Additionally, LIME does not capture non-linearities and heteroscedastic noise, which are prevalent in genomics data. To address these shortcomings, SQUID [34] was recently proposed to model the DNN in local regions of the sequence space using latent phenotype models (via MAVE-NN [101]), fitted on randomly generated synthetic data surrounding an input sequence. The parameters of these local surrogates (e.g., additive genotype-phenotype maps) can then be interpreted with standard attribution methods (Section 3.4.2).

Evolution-inspired approaches

Evolutionary algorithms (EAs) present an alternative approach for xAI, employing optimization techniques inspired by natural evolution to iteratively improve solutions over generations. In this context, a solution may constitute an explanation for a DNN prediction, which represent evolved interpretable feature sets that mimic the complex model (as done with genetic algorithms for explaining CNNs for image classification [102]). Alternatively, EAs can generate the local synthetic dataset for local surrogates, creating instances that best cover the DNN's decision

boundary, as seen with LORE [32], a popular alternative to LIME. EAs begin with a population of candidate solutions (individuals), evaluated based on a fitness function. Selection chooses the best-performing solutions to act as parents for the next generation, with genetic operators like crossover and mutation creating offspring. This iterative process continues until a termination criterion, such as a maximum number of generations or a satisfactory fitness level, is met.

An attractive subset of evolutionary algorithms is Genetic Programming (GP) [103], which evolves functional structures like computer programs or algebraic expressions without requiring prior knowledge of the solution's structure. GP can produce both linear and nonlinear models, naturally represented in interpretable tree structures [29]. Specifically, GP encodes solutions as structured syntax trees, where variables and constants are the leaves (terminals) and various operations or functions are the internal nodes. Importantly, these functions can be domain-specific, not just arithmetic operators. One way to encode domain-specific operations is by representing individuals through the use of grammars, known as Grammar-Guided Genetic Programming (GGGP) [104] (Figure 3.3). Grammars encode expressive rules that restrict the solution structure to the desired level of abstraction, making interpretability straightforward by simply evaluating the phenotype (the resulting tree) of the solution. Although never applied to genomics, we believe that GP constrained with grammars can be a powerful framework for DNN interpretability, and will be further explored in this thesis.

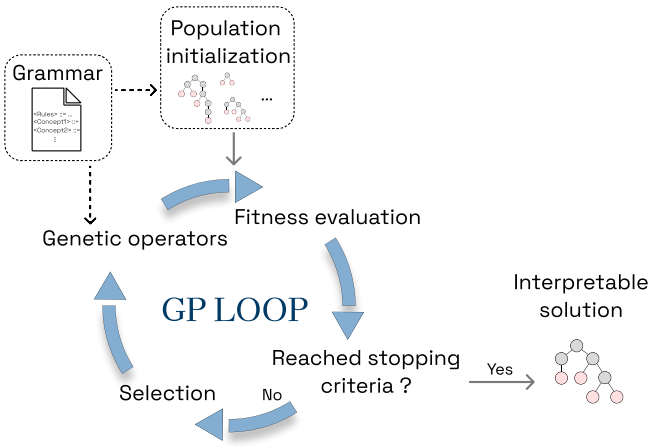

Figure 3.3: Representation of Grammar-guided Genetic Programming. The key aspect is the use of a grammar that defines the syntax of the solution space. In genomics, the grammar can encode abstractions such as motifs, motif combinations, and motif spacings. The evolution process searches for fine-grained representations within these grammatical concepts. The final program (e.g., a local surrogate model predicting the DNN score), is inherently interpretable because it adheres to the predefined grammar.

3.5 Deep learning models of RNA splicing

We can roughly divide computational models of RNA splicing into two major categories. A first group of models predicts the probability or usage of core splicing signals such as splice sites or branchpoints from primary genomic sequences. The second group focuses on predicting splicing across different conditions or tissues. Next, we introduce the type of data used to train these models, and provide an overview of models from each category.

3.5.1 Genome-wide quantification of splicing

Next Generation Sequencing (NGS) technologies enable large-scale transcriptome analysis. RNA-Seq, a widely used transcriptome sequencing approach, provides an overview of mRNA types and abundances in a population of cells. Sample size varies with sequence pool complexity and can include millions of sequencing fragments [105], which are then sequenced into short *reads*.

The RNA-Seq analytics pipeline involves mapping these short reads to a reference genome to quantify gene expression levels and alternative splicing patterns. While quantifying splicing at the gene isoform level is possible, reconstructing full splicing isoforms from short RNA-seq reads is challenging [106]. Instead, researchers often employ a more practical event-based approach, which measures binary (or more complex) splicing changes at the exon level.

Splicing outcomes are expressed as percent-spliced-in (PSI) ($\psi \in [0, 1]$), which quantifies an exon's relative inclusion level by measuring the ratio between reads supporting inclusion and the sum of reads supporting inclusion and exclusion. A PSI value of 1 indicates constitutive inclusion, while 0 indicates total exclusion. Differences in PSI are expressed using delta PSI (dPSI) ($\Delta\psi \in [-1, 1]$). Because RNA-Seq data exhibits heteroskedastic noise, contains systematic sampling bias (e.g., varying sequencing depth), and displays positional and sequence biases [107], specialized software is required to accurately estimate sample PSIs and dPSIs across conditions [108–110]. Downstream analyses typically examine exons with significant dPSI changes across conditions or tissues, which are often associated with the biological phenotype of interest.

Classical experiments to study splicing regulation often involve a gene knockdown to silence or reduce expression of a target gene. By knocking down an RBP and performing RNA-Seq on these cells, researchers can compare their splicing patterns with control samples. This allows the identification of exons with altered splicing patterns due to the RBP knockdown. Additionally, the binding profile of the RBP can be characterized through CLIP-based technologies [111], such as eCLIP [112]. The data generated identifies regions of the genome that the RBP binds to. By correlating these binding sites with the splicing outcomes measured in the knockdown RNA-Seq experiments, researchers can learn how the RBP modulates splicing [113].

3.5.2 Predicting core splicing signals

Several sequence-based strategies to predict splicing elements have existed for quite some time. Early approaches relied on principles such as maximum entropy modeling [114], PWMs [115], Markov models [116, 117], or k-mer-based features [118, 119] to score splicing elements. Most initial CNNs framed splice site prediction as a binary task, using architectures with few convolutional layers - SpliceFinder, for instance, used just one convolutional layer [120] - and typically accepted short input sequences [121, 122], or split the splice acceptor and donor predictions into separate models [122, 123]. More recently, Spliceator extended these concepts to predict splice sites across multiple species within the eukaryotic domain [124].

SpliceAI

In 2019, a landmark paper introduced SpliceAI, an end-to-end approach for splice site prediction [38]. Given its significance to this thesis, we describe this model in greater detail. Unlike previous methods, SpliceAI employs residual blocks [125] for computer vision tasks. Residual blocks in SpliceAI consist of BatchNorm layers, ReLU activation functions, and convolutional layers organized with skip connections (Figure 3.4). These skip connections enable the training of very deep networks by mitigating the vanishing/exploding gradient problem, thereby maintaining gradient flow during backpropagation and allowing for faster convergence [125].

Additionally, the convolutional units within each residual block apply increasingly larger filter sizes and dilation rates, which capture long-range dependencies in the sequence. The architecture of SpliceAI is designed to evaluate each position using a 10kb nucleotide sequence context, significantly larger than that used by any other splicing model. The output is a sequence of predictions representing the probabilities of each position being a splice donor, acceptor, or neither.

SpliceAI was trained on raw transcript sequences, with labels indicating splice site positions extracted from GENCODE annotations [126] and Genotype-Tissue Expression (GTEx) RNA-Seq data [127]. SpliceAI demonstrated remarkable results in predicting genetic variants, particularly *de novo* variants affecting cryptic splice sites. This achievement presented, for the first time, realistic hope that a model could predict clinically relevant deep intronic mutations affecting splicing [38].

Inspired by SpliceAI, Splam [128] differs in its number of residual blocks and modified convolutional layer parameters. It is designed for efficiency, using much smaller input contexts (800bp). Donors and acceptors from splicing isoforms were trained in pairs, with 200bp on each side of each splice site concatenated to form the input sequence. Additionally, the authors suggest that Splam's splice site scoring can correct errors in spliced RNA-Seq alignments. Unfortunately, the model does not predict variant effects.

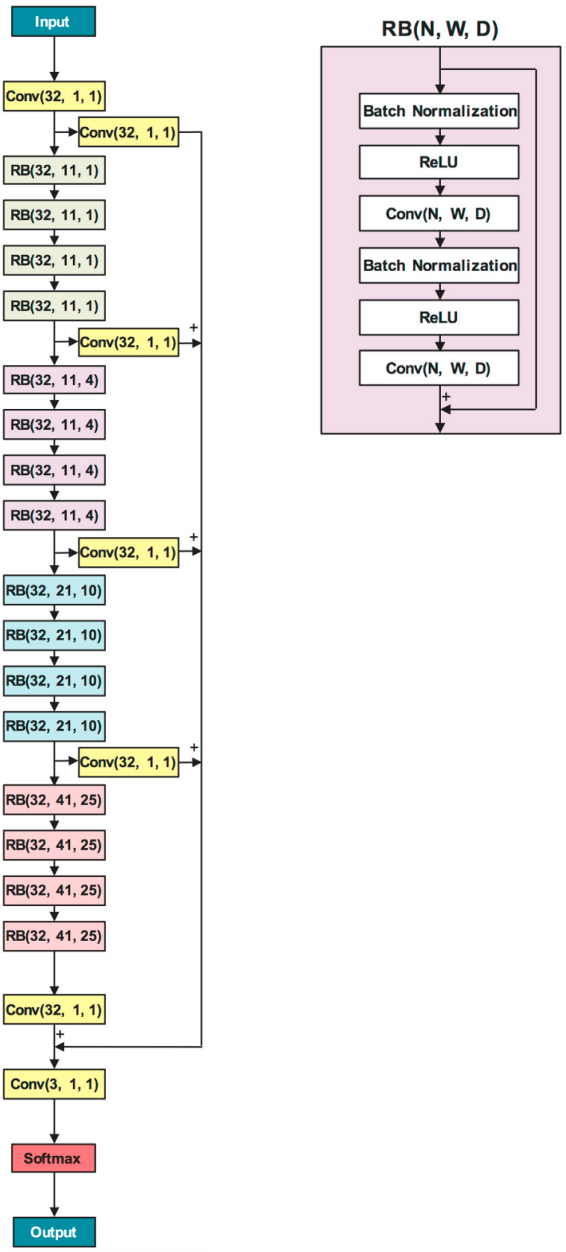

Figure 3.4: SpliceAI's model architecture. Each Residual Block, termed 'RB', employs the same number of filters 'N' in each convolutional layer (N=32), but the filter size 'W' and dilation rate 'D' increase with each block. A final layer applies the softmax function to output the probabilities of each position being a splice donor, acceptor, or neither. Figure adapted from [38].

LaBranchoR

Besides splice sites, another crucial splicing element is the branchpoint (Figure 2.1). While non-deep learning models exist for scoring branchpoints [129–131], LaBranchoR [132] is the

state-of-the-art deep learning model for branchpoint prediction.

LaBranchoR predicts splicing branchpoints directly from raw RNA sequences located 1 to 70bp upstream of genome-wide annotated 3' splice sites. It employs a two-layer Bi-directional Long Short Term Memory (Bi-LSTM) neural network with 32 hidden nodes in each direction per layer. Per base predictions are computed by taking a linear combination of the hidden states of the second layer and applying a sigmoid function to return the probability of each input position being a branchpoint site.

Models quantifying ψ

Unlike SpliceAI and earlier splice site predictors, which output probabilities for a sequence or position being a splice site, other models predict quantitative readouts of splicing outcomes. For instance, COSSMO [133] predicts ψ values from sequences by modeling the competition between alternative splice sites (e.g., 5'SS) given a fixed constitutive splice site (3'SS). The model outputs a discrete probability distribution over the alternative sites, indicating the frequency with which they are selected, serving as a proxy for ψ.

MMSplice [134] is a modular framework that predicts various components of splicing. It leverages multiple individual ANNs that score elements such as splice sites (trained from splice junctions from GENCODE), or ψ values using exon and intron modules trained with synthetic sequences from a Massively Parallel Reporter Assay (MPRA) [119]. Several regression models combining individual modules were then designed to predict variant effects on ψ and splicing efficiency, while a logistic regression classifier was used to predict pathogenicity.

3.5.3 Tissue-specific models

This category of approaches, under the umbrella of *splicing codes*, models splicing outcomes across multiple conditions, such as tissues. Early models used handcrafted splicing features from genomic sequences of cassette exons and their flanking introns [135, 136]. Due to the noisy nature of splicing quantifications (at the time using microarray data), modeling ψ or $\Delta\psi$ as continuous random variables was challenging. Thus, the task was formulated to predict the probability of differential splicing between tissue pairs.

Leung *et al.* [137] and Xiong *et al.* [138] used a fully connected neural network and a Bayesian Neural Network, respectively, to improve over the previous methods by incorporation of splicing quantifications from RNA-Seq data to predict real-valued absolute ψ values in more tissues. The latter, called SPANR [138], was a pioneering effort in quantifying tissue-specific effects of genetic variants on splicing. A later attempt tried to incorporate CLIP data into these models, but the results were modest [139].

With the advent of CNNs and large-scale tissue-specific RNA-Seq data (e.g., GTEx consortium), new models trained directly from raw sequences emerged. MTSplice extends the

MMSplice framework to quantitatively predict tissue-specific variant effects [140], proposing a new CNN (TSplice) that predicts ψ values across 56 tissues using a multitask learning approach.

Pangolin [141] leverages the SpliceAI architecture, with slight modifications in the last two layers, to predict both splice site usage and probability from primary sequences using tissue-specific RNA-Seq data from multiple species.

Recently, the AbSplice framework [142] was developed to predict tissue-specific splicing outliers (large $\Delta\psi$ changes) caused by genetic variants. The authors created tissue-specific splice site annotations (SpliceMaps) and combined them with SpliceAI and MMSplice to build a generalized additive model, providing probability estimates of aberrant splicing in a given tissue of interest.

3.5.4 Summary

This section provided an overview of deep learning models applied to RNA splicing, highlighting key conceptual advances. Early in this PhD work, we confirmed the potential of deep learning to score clinically relevant intronic variants. In a large cohort of Hypertrophic Cardiomyopathy patients, we identified cryptic splice site variants and a branchpoint-disrupting variant in the *MYBPC3* gene using SpliceAI and LaBranchoR, respectively, which co-segregated with disease phenotypes in families [35].

Next, we will benchmark these models, along with others focused on interpretability, for predicting disease-associated deep intronic variants across various diseases, sequence contexts, and splicing defects. This task serves as a challenging test to assess the generalization capacity of these models and their understanding of the splicing mechanism.

Chapter 4

Predicting intronic variants affecting the splicing mechanism

This chapter presents a benchmark of the state-of-the-art methods for predicting and interpreting deep intronic variants that disrupt RNA splicing. Towards this goal, we first provide a comprehensive overview of VEPs, their scope and limitations and then we developed a software to streamline VEPs benchmarking. Moreover, we curated several relevant datasets for the task, including a new dataset of deep intronic variants causing human disease and region-specific variant sets disrupting splicing through different molecular mechanisms. This work gave rise to multiple contributions:

- Development of VETA, a software for benchmarking variant prediction tools.
  Code: https://github.com/PedroBarbosa/VETA
  Paper: **P. Barbosa**, M. Ribeiro, M. Carmo-Fonseca, A. Fonseca, "Clinical significance of genetic variation in hypertrophic cardiomyopathy: comparison of computational tools to prioritize missense variants", *Frontiers in Cardiovascular Medicine*, 2022 [36].
  Reproducibility: https://github.com/PedroBarbosa/paper_HCM_benchmark
- Manual curation of splicing-disrupting benchmark datasets
  DOI: https://doi.org/10.5524/102423
- A simple tool that processes VCF files and generates the correct input to run several sequence-based splicing predictors.
  Code: https://github.com/PedroBarbosa/Prepare_SplicingPredictors
- Benchmarking performance and interpretability of VEPs in deep intronic regions.
  Paper: **P. Barbosa**, R. Savisaar, M. Carmo-Fonseca, A. Fonseca, "Computational prediction of human deep intronic variation", *GigaScience*, 2023 [37].
  Reproducibility: https://github.com/PedroBarbosa/DeepIntronic_Benchmark

4.1 Introduction

Genetic variation plays a crucial role in understanding human disease and trait inheritance. Yet, for a long time, studies paid scant attention to variants in intronic gene regions [143], which were thought to harbor little functional variation. With the widespread adoption of WGS, intronic variants can be identified more effectively than ever before. Nevertheless, the large number of candidate variants discovered brings significant challenges for their functional interpretation [16], especially for those affecting RNA splicing [17].

Given the challenges of interpreting deep intronic mutations (previously discussed in Section 2.1.2), computational tools are often used to prioritize variants based on their likelihood of being deleterious. The first wave of methods used large genomics datasets to engineer features (e.g., allele frequencies from ExaC [144] or histone modification levels across cell lines from ENCODE [145]) and to build classifiers that work on tabular data. More recently, end-to-end deep-learning methods predict the impact of genetic variants from sequence alone, with the features automatically extracted within the network [76]. SpliceAI [38] is widely recognized as the most successful method of this kind, although its performance has been shown to vary across studies and datasets considered [17]. Recently, new models have been developed based on SpliceAI, either combining its predictions with other sources of information (such as genetic constraint for ConSpliceML [146] and PDIVAS [147], or tissue-specific splice site usage for AbSplice-DNA [142]), or even creating an entirely new model based on SpliceAI architecture. For example, Pangolin [141] uses splicing quantifications from multiple species and tissues to not only predict whether a position is a splice site (as SpliceAI does) but also to predict splice site usage (e.g., how much a splice site is being used in a given tissue). In contrast, CI-SpliceAI [148] uses different training labels for true and false splice site positions based on a collapsed transcript structure derived from GENCODE [126] annotations.

Most intronic variant prediction benchmarking studies are performed by the authors of the tools to present a comparative analysis with existing methods. Even subconsciously, biases might be favoring the proposed model, be it because of the dataset selected or the methodology employed for the comparison [149, 150]. Multiple independent benchmark studies do exist [151–157], however, their scope is often somewhat limited. Firstly, some studies only focus on variants overlapping particular types of splicing information, e.g. splicing regulatory elements [151, 153]. Secondly, only using variants from a few genes can render the genome-wide extrapolation of conclusions difficult [152, 154, 156]. Lastly, to our knowledge, no study compares the performance of promising and recently developed methods such as Pangolin, CI-SpliceAI, ConSpliceML, AbSplice-DNA, PDIVAS and SPiP [158].

To help researchers and clinical practitioners understand prediction tools and how they can be applied to interpret genetic variants in introns, we conducted a comprehensive evaluation of a series of tools for the task of predicting functional variation in the intronic space far

from canonical splice sites (Sections 4.4 to 4.6). To this end, we carefully selected intronic variants from multiple sources and curated a new set of disease-causing deep intronic variants affecting RNA splicing (Appendix A.1). Besides evaluating the capacity of tools to predict functional variants deep within introns, we also report, for the first time, an assessment of the interpretability of these tools' outputs (Section 4.7) and their capacity for accurate tissue-specific variant prediction (Section 4.8). We finally provide clear recommendations for tool usage depending on the variant's location within the intron and its molecular effect (Section 4.9).

4.2 The prediction tools studied are diverse in methodology and objectives

We now provide a snapshot of the state-of-the-art of methods that predict, in any way, functional variation in introns (Table 4.1, details on how we annotated VCF files with VEPs scores in Appendix A.2). We divided the methods into four different categories: conservation scores that measure the degree of evolutionary conservation at a given position or region of the genome; genome-wide predictors that integrate multiple feature types to predict variant effects regardless of the variant type; methods that focus on splice-disrupting variants and allow for automated batch predictions; and splicing-specific methods that solely target specific types of splicing information (e.g., BP), or require the use of a web application to retrieve results. For many tools, there are two fundamentally different ways to obtain predictions: making *de novo* model inferences given an input variant set or using pre-computed predictions, which is faster computationally. We decided to use pre-computed predictions when available because it considerably simplifies the variant annotation pipeline and is thus accessible to a more diverse set of end users. However, it should be noted that this approach may miss some indels that are not represented in the pre-computed databases. Of the 38 tools used to score at least one dataset in this study, 19 had pre-computed databases available (Table 4.1). Because some of them only provide predictions for the GRCh37 genome build, we ran all experiments using this genome version. Of note, pre-computed predictions are a permanent representation of a model version, which may not be updated along with developments to the tool. However, we observed that only one tool, CAPICE [159], had outdated pre-computed scores. Importantly, not every tool considered was built with deep intronic regions in mind. For example, some tools were explicitly trained only to score consensus splice site variants (e.g. MaxEntScan [114], dbscSNV [160]), while others only output predictions up to an approximately defined distance between the variant and the nearest splice site (e.g., 300 bp for SPIDEX [138] or 50 bp for MLCSplice [161]). In addition, we ran certain models (KipoiSplice4 [82], HAL [119], MMSplice [134]) using the Kipoi framework [82], which further restricts predictions to a tool-specific distance between the splice site and the variant. Therefore, we expected these methods to perform poorly on some

comparisons simply because the fraction of missing predictions should increase when moving further into the intron. Still, we decided to include these tools in the study because many of the variants evaluated locate within the distance that we expected these tools to cover.

It should also be noted that the tools were built for different tasks. While some models were designed to distinguish between pathogenic and benign variants (e.g., S-CAP [162], KipoiSplice4), others predict variant effects on splicing outcome, which does not necessarily translate into disease (e.g., SPiP, MMSplice). The latter category comprises sequence-based deep learning models such as SpliceAI or Pangolin. While these packages accept genetic variants in VCF format as input, it is important to note that the models primarily operate on sequences. They predict the probability of a given sequence position functioning as a splice site. If the model is run twice, once with the reference and once with the mutated sequence, it is possible to assess splice site alterations caused by genetic variants with the so-called delta score (mutated - reference allele). This has been the major practical use of the tool so far. Using the same rationale, we also included several sequence-based methods that predict splicing-related elements. These include SpliceRover [123], DSSP [122], and Spliceator [124] for splice site-associated variants, ESEfinder [163], ESRseq [118] and HEXplorer [164] for variants affecting splicing regulatory elements, and SVM-BPFinder [129] and BPP [130] for variants impacting the BP signal. Of note, we only employed these methods for the datasets deemed to be relevant given their original task.

4.3 Enabling variant effect prediction benchmarking with VETA

VEPs are highly heterogeneous, with their input requirements and output formats varying significantly. In addition, the number of VEPs is continuously increasing, making it challenging to keep track of their underlying methodologies, the interpretation of the output score and decision threshold for fair assessment against other methods [165]. Therefore, we developed Variant prEdiction Tools evAluation (VETA), with the goal of simplifying benchmarking of VEPs and standardizing such an important aspect of variant interpretation. VETA has embedded support for more than 50 VEPs, yet users can easily add custom methods not included by default. Figure 4.1 provides a high level overview of VETA workflow.

Table 4.1: Summary of the computational methods used in this study.

	Tool *	Threshold †	Description	Method	Training data	Predictions from ‡	Used in analysis **
Conservation	phastCons 100way [166]	> 0.99 [167]	Probability that each nucleotide belongs to a conserved element	Hidden Markov Model	Genomes of 100 vertebrates	Pre-computed (UCSC)	ClinVar
	phyloP 100way [168]	> 1.6 [169]	P-value that indicates how aligned sequences deviate from the null hypothesis of neutral evolution	Hidden Markov Model	Genomes of 100 vertebrates	Pre-computed (UCSC)	ClinVar
	SiPhy 29way [170]	> 12.7 [169]	Identification of constrained sites as those with a nucleotide substitution pattern significantly deviating from the neutral pattern	Maximum Likelihood and Hidden Markov Model	Genomes of 29 mammals	Pre-computed (dbNSFP)	ClinVar
	GERP [171]	> 4.4 [169]	Identification of evolutionarily constrained elements	Maximum Likelihood to estimate the evolutionary rate and dynamic programming	Genomes of 34 mammals	Pre-computed (UCSC)	ClinVar
Genome-wide predictors	FATHMM-MKL [172]	> 0.5 [173]	Prediction of functional consequences of coding and non-coding SNVs using genomic annotations from ENCODE and conservation scores	Support Vector Machine based on Multiple Kernel Learning	3,063 disease-implicated SNVs from HGMD; 5,252 negative instances from 1000G project [174]	Pre-computed (dbNSFP)	ClinVar
	Eigen v1.1 [175]	> 4.87 [162]	Unsupervised learning approach to leverage the functional importance of genetic variants across the whole genome	Linear combination of the components of the leading eigenvector determined from a rank-one matrix estimated from 3 genome-wide annotation blocks.	418,997 variants from 1000G project	Pre-computed (dbNSFP)	ClinVar
	ReMM v0.3.1 [176]	> 0.984	Classifier to predict the potential of an arbitrary position in the genome to cause a Mendelian disease	Random Forest	453 disease-implicated variants by manual curation	Pre-computed (tool webpage)	ClinVar
	LINSIGHT [177]	> 0.056 [162]	Prediction of non-coding nucleotide sites at which mutations are likely to have deleterious fitness consequences	INSIGHT and Online stochastic gradient descent	Genomes of 54 unrelated human individuals	Pre-computed (dbNSFP)	ClinVar
	CAPICE v1.0 [159]	> 0.02	A consequence-agnostic method for pathogenicity prediction	XGBoost	Data from ClinVar, VKGL [178] and specific publication	Pre-computed (Zenodo)	ClinVar
	CADD-Splice v1.6 [179]	> 15 [169]	Prediction of the deleterious effect a variant has on an individual's fitness	Logistic Regression	16,627,775 of both proxy-neutral and proxy-deleterious variants	Pre-computed (tool webpage)	ClinVar; Splicing Pathogenic
	MaxEntScan [114]	$\lvert\Delta Entropy\rvert > 3$	Prediction of RNA splice site signal based on the maximum entropy principle	Maximum Entropy distribution	8,500 real 5'SS and 3'SS; 180,000 decoy 5'SS and 3'SS	VEP plugin [180]	ClinVar; AU; NSD
	dbscSNV v1.1 [160]	> 0.6	*In silico* prediction of splice-altering variants based on an ensemble of individual methods	AdaBoost and Random Forest	Splice-altering variants from HGMD, SpliceDisease [181] and DBASS [181] databases. Negative variants from 1000G Project	Pre-computed (dbNSFP)	ClinVar; Splicing Pathogenic
	SPIDEX v1.0 [138]	$\lvert\Delta PSI\_zscore\rvert > 2$	Prediction of how much SNVs cause splicing misregulation by measuring differential exon inclusion events	Bayesian Deep neural network	RNA-Seq data in 10,700 exons across 16 tissues	Pre-computed (tool webpage)	ClinVar; SplicingPathogenic; BP
	HAL [119]	$\lvert\Delta PSI\rvert > 0.05$§	Variant effect prediction on different isoform usage from alternative splicing events (alternative 5'ss and exon skipping)	Linear model using hexamer motif frequencies	MPRA containing 265,137 minigenes in a library of alternative 5' splice donors	Kipoi (only 5'ss model)	ClinVar; NSD; DD

Splicing	TrAP v3.0 [182]	> 0.174	Prediction of the damage caused by SNVs at the transcript level by incorporation of splicing-engineered features	Random Forest	75 pathogenic synonymous variants; 402 synonymous variants as benign	Pre-computed (tool webpage)	All
	S-CAP v1.0 [162]	Several thresholds ‖	Splicing-specific pathogenicity score derived from variant, exon and gene importance measurements	Gradient Boosting tree	17,059 splicing-related pathogenic variants from HGMD and Clinvar and 6,760,450 splicing region benign variants from gnomAD	Pre-computed (tool webpage)	ClinVar; SplicingPathogenic; BP
	KipoiSplice4 v0.1 [82]	> 0.5	Ensemble method that incorporates predictions from 4 splicing-related models (HAL, MaxEntScan5, MaxEntScan3 and LaBranchoR)	Logistic Regression	10,715 splice region variants from Clinvar and 2,959 variants from the dbscSNV paper [160]	Kipoi	ClinVar; SplicingPathogenic; BP
	SpliceAI v1.3 [38]	> 0.2	Splice site prediction from primary sequence	Deep residual neural network	Primary transcript of 13,384 genes, accounting for 130,796 donor-acceptor pairs, plus novel splice junctions observed in GTEx data [127]	Pre-computed (tool webpage)	All
	MMSplice v1.03 [134]	$\|\Delta logitPSI\| > 1$	Modular approach to study functional effects of variants on splicing	Linear model that combines coefficient of 5 neural network modules	MPRA (Vex-Seq) designed to evaluate the effect of 2059 ExAC variants on exon skipping #	Kipoi	ClinVar; SplicingPathogenic; BP
	SQUIRLS v2.0.1 [183]	> 0.074 [148]	Prediction of the effect of variants on splicing providing interpretable outputs	Logistic regression model combining predictions from two Random Forests classifiers (donor and acceptor)	Cytoband-aware split of 73,203 benign variants from ClinVar and 8,314 deleterious variants from ClinVar and manual curation of variants from literature	Model inference	All
	Pangolin v1.02 [141]	> 0.2	Splice site prediction from primary sequence across multiple tissues	Deep residual neural network	Sequences and splice site quantifications from four species: human, rhesus macaque, rat and mouse	Model inference	All
	CI-SpliceAI v1.0 [148]	> 0.190	Same as SpliceAI	Deep residual neural network	Sequences of 18,580 genes with splice sites (428,275) collapsed from GENCODE isoforms	Model inference	All
	ConSpliceML v0.0.6 [146]	> 0.5	Combination of SpliceAI and SQUIRLS predictions along with a metric of genetic constraint against deleterious splicing variation	Random Forest	18,317 splicing-altering HGMD variants plus benign de novo variants collected from whole genome sequencing studies and gtex	Pre-computed (tool webpage)	All
	AbSplice-DNA v0.0.1 [142]	> 0.01	Aberrant splicing prediction using MMSplice, SpliceAI and tissue-specific annotations derived from GTEx	Generalized additive model	Splicing outliers detected from 946 GTEx individuals with paired RNA-Seq and WGS data	Pre-computed (Zenodo)	All
	MLCsplice [161]	> 0.5	Meta-predictor incorporating multiple splicing-related scores to predict region-specific variants	Hybrid model based on XGBoost, CGBoost and LightGBM	Positive variants obtained from DBASS and HGMD database. Negative variants retrieved from gnomAD, ExAC and dbSNP [184] with MAF $> 10\%$	Pre-computed (tool webpage)	ClinVar; SplicingPathogenic; BP
	SPiP v2.1 [158]	> 0.452	Prioritization of splicing variants by running complementary bioinformatic tools that model different splicing elements	Random Forest	Random 50% split of 4,416 curated splicing-altering variants and 95,000 control variants	Model inference	All
	PDIVAS v1.0.0 [147]	> 0.151	Pathogenic prediction of deep intronic variation combining SpliceAI (including raw scores), MaxEntScan and ConSplice features	Random Forest	374 pathogenic variants from HGMD and [60]; 153,794 benign variants from the 1000G project	Model inference ‡‡	SplicingPathogenic; AU; EL; NSD; DD
	ESEfinder v3.0 [163]	$\|\Delta score\| > 0.5$ ***	Identification of exonic splicing enhancers from weight matrices of four SR proteins derived from SELEX experiments	Scoring motifs of each SR protein against a predefined threshold (inferred from high-scoring randomly chosen sequences from the initial SELEX library)	-	Webpage & Own code	EL

Splicing (Region-specific or web-based)	ESRseq [118]	$\lvert\Delta score\rvert$ > 0.5 [152]	QUEPASA, a minigene assay that measured the impact of 6-mer motifs in RNA splicing	Statistical comparison of observed splicing strengths in sequences where the motif is present vs absent	-	Own code	EL
	HEXplorer [164]	$\lvert\Delta score\rvert$ > 14 [152]	RESCUE-based approach to score elements that enhance or repress splice site usage	Average Z-score HZei (based on hexamer frequencies in exonic vs intronic sequences) of all six hexamers overlapping with any given nucleotide	-	Webpage & Own code	EL
	IntSplice2 v2.0 [185]	> 0.5	Prediction of pathogenic intronic SNVs upstream of splicing acceptors	LightGBM	1,787 of each class located at -50 to -3bp of splicing acceptors. Pathogenic variants from HGMD and ClinVar. Neutral variants from dbSNP.	Pre-computed (tool webpage)	SplicingPathogenic; BP; AU
	SVM-BPFinder [129]	$\lvert score\rvert$ > 0.136 [151]	Branchpoint prediction using sequence signals and additional polypyrimidine tract features	Support Vector Machine	Positive sequences: intronic 9-mers conserved across multiple species. Negative sequences: random intronic 9-mers. Both sets had T and A at positions 4 and 6.	Model inference & Own code (as in [151])	BP
	BPP [130]	$\lvert score\rvert$ > 0.0006 [151]	Branchpoint prediction using sequence features extracted from conserved intronic regions of the human genome	Mixture model to predict branchpoint motif combined with octanucleotide frequencies in PPT region	223,606 human introns longer than 300bp	Model inference & Own code (as in [151])	BP
	LaBranchoR [132]	$\lvert\Delta score\rvert$ > 0.1	Prediction of splicing branchpoint signals from raw sequence	Bi-LSTM neural network	Highly confident branchpoints that matched GENCODE-annotated 3'ss	Kipoi	BP
	BPHunter v2[131]	> 1 ††	Detection of intronic variants that disrupt the branchpoint sequence	Integration of Gradient Boosting tree, Random Forest and Logistic Regression with the majority voting for the final prediction	198,256 branchpoint positions with flanking 13-bp and 1 million 13-bp random intronic and exonic positions	Webpage & Own code	BP
	SpliceRover [123]	> 0.5 ***	Splice site prediction from primary sequence	Convolutional neural network	Sequences of arabidopsis and human surrounding canonical splice donors and acceptors	Webpage & Own code	AU; NSD; DD
	DSSP [122]	> 0.5 ***	Prediction of the impact of SNVs on splicing using a combination of deep learning and standard machine learning with handcrafted features	Stack generalization to combine a convolutional neural network with a Random Forest, XGBoost and Linear Regression models	170-bp sequences representing 4,964 variants (with corresponding wildtype sequences) from the MaPSy experiment [186]	Model inference & Own code	NSD
	Spliceator v1.0 [124]	> 0.5 ***	Splice site prediction for multi-species data	Convolutional neural network	Sequences from multiple species, from protists to human	Model inference & Own code	AU; NSD; DD

*We refer to the specific tool version used, although for several tools we did not find a reference pointing to any version.
†Cutoff used to discriminate pathogenic/functional variants. If the original paper did not provide a reference threshold, it was extracted from elsewhere, with another reference assigned.
‡ 'Own code'refers to our package [187].
** Analysis where tool was used. Where acronyms are seen, it refers to the region-specific splicing analysis: BP = Branchpoint-associated; NSA = New splice acceptor; NSD = New splice donor; AU = Acceptor upstream; DD = Donor downstream; EL = Exonic-like.
§ HAL scores PSI for the sequence containing alternative 5'ss variants. Therefore, for this work, a change in PSI > 0.05 was defined as the relevant threshold.
‖ S-CAP authors provide different reference thresholds depending on the location and context of the variant.
Several models were fitted in MMSplice. In the table, the details of a single model are provided, the one that predicts $DeltalogitPSI$ changes, as it was the primary goal defined.
*** When no reported threshold was found, we set 0.5 as the default value.
†† BPHunter threshold adjusted to 1 after discussing with the tool's author. Annotated variants with 0 score are shifted to 1, and all unannotated variants are assigned a score of 0.
‡‡ PDIVAS does provide pre-computed scores, but those only include pathogenic predictions. To get scores for variants that are not predicted as pathogenic, we need to perform raw model inferences.

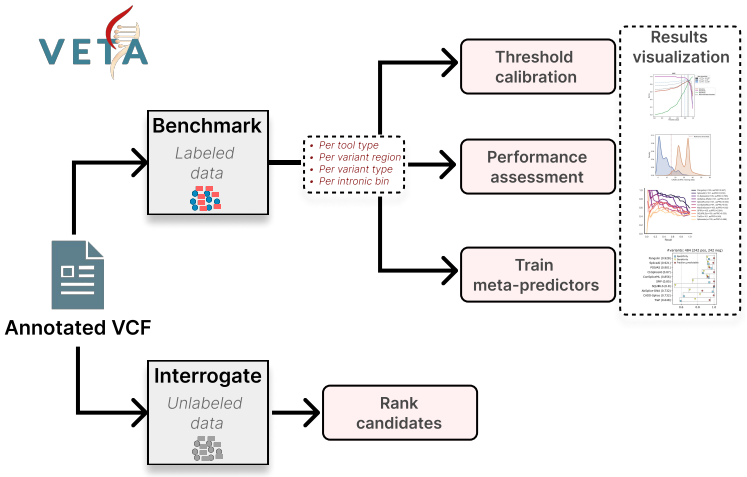

Figure 4.1: VETA high level workflow. VETA has two main running modes, `benchmark` and `interrogate`. The former assumes labeled data and benchmarks VEPs based on that information. The latter ranks variants according to the consensus among predictive tools regarding pathogenicity. In either mode, VETA takes care of harmonizing and standardizing the diverse set of VEPs outputs, rendering them ready for the analysis.

Briefly, VETA takes as input labelled VCF files (e.g., benign and pathogenic variants) annotated with ensembl VEP [188], the most used variant annotator. This single requirement has the main purpose of enabling more fine-grained analyses, depending on the user needs. For example, the VCF may contain variants occurring at different genomic locations (e.g, coding, intronic, untranslated regions (UTRs)). By taking advantage of ensembl VEP annotations, VETA can automatically stratify benchmarks for each region, where performance may differ significantly [189]. Moreover, VETA includes evaluation metrics that deal with different scenarios such as class unbalancing, the lack of decision thresholds or low coverage (high rate of missing predictions).

4.3.1 Threshold calibration

One import aspect for using VEPs in variant interpretation for clinical purposes is determining the decision threshold at which a variant is considered to harbor evidence towards pathogenicity. Often, tool authors do not provide a clear threshold, of if provided, it may be inadequately calibrated. VETA offers a straightforward way for deriving an optimized threshold, allowing for different levels of importance given to precision and recall. In short, for each tool VETA calculates the F-Beta score across 100 thresholds uniformly distributed over the observed range of scores, according to the formula:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{Precision} \cdot \text{Recall}}{(\beta^2 \cdot \text{Precision}) + \text{Recall}} \tag{4.1}$$

where β is a parameter allowing to balance the weight given to precision ($TruePositives(TP)/\ TP+FalsePositives(FP)$) and recall ($TP/TP+FalseNegatives(FN)$). VETA tests three β values by default: 0.5, 1 and 1.5. For each β value, the threshold that maximizes the F-Beta function is selected as calibrated threshold. Higher β values favor sensitivity, allowing to capture more pathogenic variants at the cost of increasing false positives. Conversely, lower β values favor precision, at the cost of possibly missing some true positives.

The reliability of the adjusted thresholds can be further examined through bootstrapping. For each tool, 1000 bootstrap samples are generated, ensuring the same class ratio as in the original dataset. Then, the best threshold is derived for each bootstrap sample as described above with the F_β formula. Finally, the 0.025 and 0.975 quantiles of the distribution of the bootstrap sample statistic (which is a distribution of the best thresholds) are computed. These values are then used to interrogate at which threshold range 95% of the bootstrap sample statistic lies, and how wide/narrow this interval is in respect to the threshold obtained with the true data.

4.3.2 Combining tools into meta-predictors

Many VEPs incorporate predictions from other VEPs as features within their models (e.g, [142, 146, 147, 161]). They are called meta-predictors, and the rationale behind them is that exploiting and combining the strengths of different methods can lead to more robust results. VETA has a module that combines predictions of multiple tools and creates meta-predictors using standard scikit-learn models [190]. Importantly, VETA automatically streamlines hyperparameter optimization via grid search, evaluates the models with stratified cross-validation, and compares performance of the meta-predictors with individual counterparts.

These utilities serve as an initial step in assessing the potential advantages of combining multiple methods. Accordingly, VETA seamlessly integrates feature importance analysis (e.g, information gain, recursive feature elimination) to study which tools are more likely to provide complementarity to the meta-predictor.

4.3.3 Faithful benchmarking of intronic variants

VETA is designed to handle the specificities of intronic variants. For instance, it is expected that variants located at the splice site dinucleotide are likely to disrupt splicing and therefore VEPs, trained with this information, should perform well. However, the further the variant is from the splice site, the more challenging it may be to predict its effect. VETA stratifies the benchmark in bins, according to the distance between the variant and the nearest splice site. As it will be discussed in the next section, this feature was crucial for more realistic evaluation of the tools. To add on that, it is possible to do separate benchmarks for variants closer to acceptor and donor splice sites, which may also be of great utility.

4.3.4 Availability

VETA is a versatile software that can be used to benchmark VEPs in a standardized way. Moreover, it includes an `interrogate` mode that ranks variants according to how much VEPs agree on their evidence for pathogenicity, regardless of existing labels. It is of practical use in whole-genome and exome sequencing studies to help narrow down candidate variants for further validation.

The software, which is open source and written in Python, features an intuitive command line interface, and is thoroughly documented on its GitHub repository (`https://github.com/PedroBarbosa/VETA`). VETA is stored in the Python Package Index (PyPI), and also accessible via a pre-built Docker image (`https://hub.docker.com/r/pbarbosa/veta`).

4.4 ClinVar variants located beyond 10 bp from the splice sites are poorly predicted

Having introduced VETA, we now present the results of the benchmark. First, we employed a bin-based analysis to evaluate ClinVar data (full dataset is available at `https://s3.ap-northeast-1.wasabisys.com/gigadb-datasets/live/pub/10.5524/102001_103000/102423/supplementary_tables/table_S1_clinvar_data.tsv`). We evaluated performance using a variation of the F1 score called the weighted F1 score. This metric considers missing predictions during evaluation (refer to Appendix A.3 for details on the metrics used). Because ClinVar contains disease-causing variants that act through different molecular mechanisms, we included not only splicing-related tools but also conservation scores and whole genome predictors in the evaluations. Some of the models were trained using ClinVar data (Table 4.1), potentially leading to a circularity type I problem [191]. Fully correcting for this issue would have signified removing all ClinVar variants that were used in the training of any of the tools. This would have been problematic, as we would have lost many valuable deep intronic variants, which are typically scarce. However, most of the tools that were trained with ClinVar variants performed poorly. CAPICE was the only one to achieve a weighted F1 score above 0.6 across all bins (Figure A.1A). We therefore only removed ClinVar variants that were used for training CAPICE (N=14,189: 5,205 pathogenic and 8,984 benign). This is a trade-off, allowing for over-estimated performance for some of the more underperforming tools while ensuring a sufficiently large dataset for the evaluation of all tools. After this filtering step, 53,600 variants remained for evaluation. As expected, the distribution of the two variant classes (pathogenic and benign) across bins is highly unbalanced (Figure 4.2A). More than 90% of the intronic pathogenic variants occur at splice site positions, and more than 95% occur within 10 nucleotides from an exon-intron boundary.

Due to the spatial limitations discussed previously, we expected that some splicing tools

would only output predictions for ClinVar variants located close to splice sites. Our results confirmed that several methods make predictions for less than 50% of the variants located at a distance of more than 40 bp from the nearest splice junction (Figure 4.2B). The fraction of predicted variants decreases according to the expected regions that each model covers: 50bp for S-CAP and MLCsplice, and 300bp for SPIDEX. MLCsplice was designed to predict non-canonical splicing variants (i.e. excluding splice site variants), thus, it is the only tool that displays no predictions at 1-2 positions (Figure 4.2B). In addition, we observed that the tools ran using the Kipoi framework (KipoiSplice4, HAL, MMSplice) displayed a notable drop 41-200 bp from the splice site. On the other hand, SQUIRLS [183], Pangolin, CI-SpliceAI, SPiP, TraP [182], SpliceAI, ConSpliceML and AbSplice-DNA predicted across entire introns (Figure 4.2B). Regarding the remaining tool categories, we observed that both whole genome predictors and conservation scores (except phastCons [166]) output predictions for most ClinVar variants (Figure 4.2B).

Next, we evaluated how the tools that score across full introns perform with ClinVar data. Performance dropped considerably for variants located deeper in intronic regions, especially once a distance of 10 nucleotides from the splice site had been reached (Figure 4.2C). The splicing tools with the smallest and largest performance decrease between the splice site bin ("1-2") and the "11-40" bin were Pangolin and TraP, with weighted F1 scores decreasing by 0.303 and 0.757, respectively (Figure 4.2C). Conservation scores and whole-genome predictors performed poorly as well. Except for CAPICE and CADD-Splice, most methods displayed weighted F1 scores below 0.15 at the 11-40 bin. Overall, the most performant tools were CI-SpliceAI, Pangolin and SpliceAI with average weighted F1 scores across all intronic bins of 0.672, 0.661 and 0.627, respectively (Figure 4.2C). A table with all metrics for each tool and bin can be found at `https://s3.ap-northeast-1.wasabisys.com/gigadb-datasets/live/pub/10.5524/102001_103000/102423/supplementary_tables/table_S2_clinvar_data_results.tsv`.

Strikingly, we noticed an increase in performance in the deepest intronic bins when compared to intermediate distances (Figure 4.2C). We hypothesized that variability in transcript structures could be the reason: despite these variants being assigned as occurring very deep within introns ($>$ 500bp from the splice site) according to the associated RefSeq transcript, they may be exonic or near-splice site variants in other isoforms of the associated gene. To tackle this question, we looked at the raw transcript annotations (without picking ensembl VEP consequences) of the variants assigned to the $>$ 500bp bin (N=1501) and decomposed them into several sub-categories based on their localization in different transcript isoforms (see methods details in Appendix A.4). Our analysis revealed that 304 variants are located in exons in other transcripts and 274 variants mapped to introns but closer to splice sites than in the transcript isoform originally considered (Figure A.1B). In particular, some of the intronic variants are located at splice sites in other transcripts (Figure A.1C). We found that the performance of the tools was generally better for these categories than for categories where the variant distance to the splice site remained

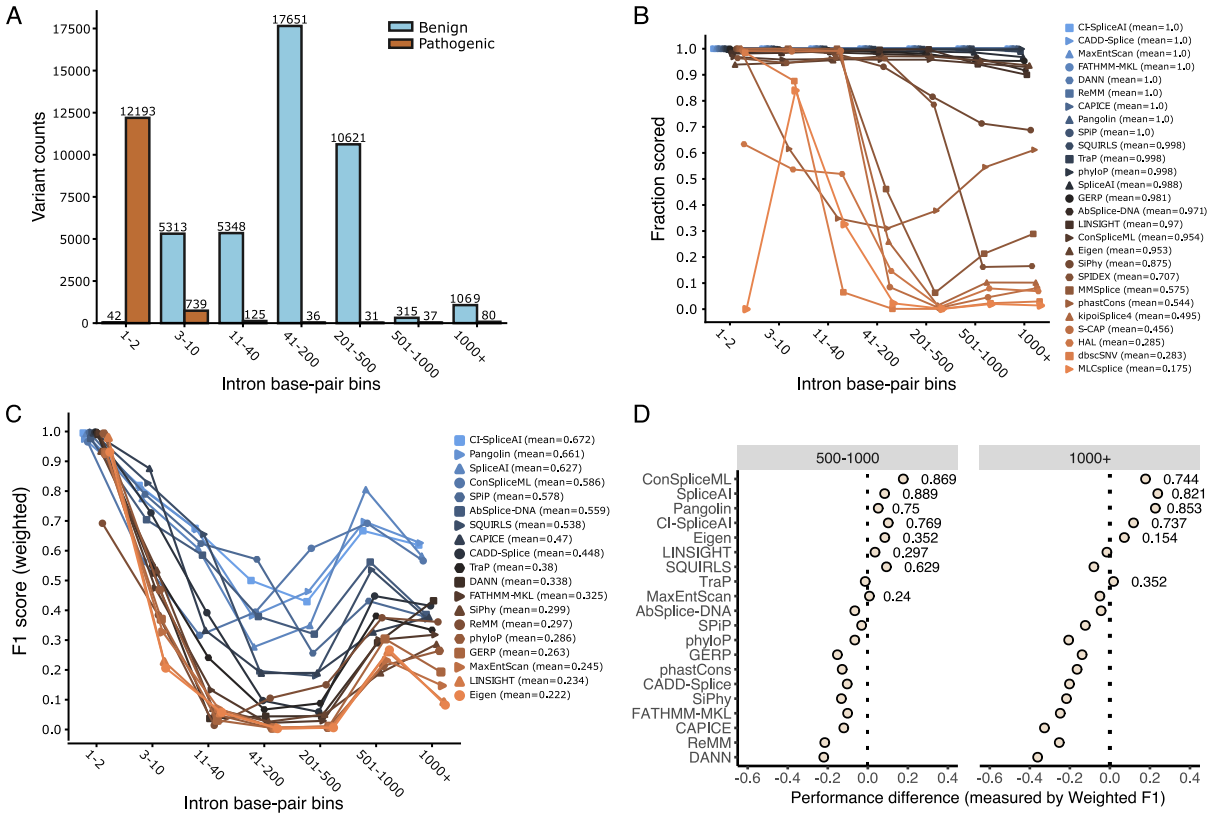

Figure 4.2: Intronic variant prediction in ClinVar. **A** - Distribution of variants across each intronic bin considering the RefSeq transcript associated with each ClinVar variant. **B** - Fraction of variants scored (with predictions) at each intronic bin. Mean values in the legend represent the average fraction of variants scored across all bins. **C** - Performance of tools that predict entire introns (defined as > 90% scored variants) at each intronic bin. Mean values in the legend represent the average weighted F1 score across all bins. **D** - Differences in performance per deep intronic bins ("501-1000" and "1000+") after removing variants that are exonic or closer to splice sites in other transcripts of the associated gene. Points refer to the weighted F1 difference between this new analysis minus the values obtained originally (displayed in C). Annotations next to points refer to the weighted F1 scores in the new analysis for the tools whose performance difference is positive.

unchanged (Figure A.1D), which is consistent with the hypothesis that deep intronic pathogenic variants are hard to predict. After excluding variants from exonic and closer-to-splice sites categories, we repeated the per-bin analysis to see whether the performance increase in the deepest bins remained. We observed that most conservation-based methods and whole genome predictors displayed a decline in performance compared to the original analysis (Figure 4.2D). On the other hand, a subset of splicing tools such as ConSpliceML, SpliceAI, Pangolin or CI-SpliceAI showed better performance than before, suggesting that unequivocal deep intronic variants in ClinVar are associated with splicing and that SpliceAI-based methods can identify them reasonably well.

4.5 Pathogenic splicing-affecting variants are captured well by deep learning based methods

Not all ClinVar intronic variants are associated with splicing defects. However, splicing-related tools were the most successful at predicting the pathogenicity of deep intronic mutations. Therefore, we decided to narrow our focus to variants that specifically affected splicing. Our lab previously published a dataset of deep intronic variants causing human disease via disruption of splicing (N=81) [59]. In the current study, we augmented the dataset by performing a comprehensive literature search for case reports published after 2017, where the association between a variant and a splicing defect was supported by experimental evidence, such as from RT-PCR, sequencing of cDNA products, RNA-Seq or minigene/midigene assays (full dataset available at https://s3.ap-northeast-1.wasabisys.com/gigadb-datasets/live/pub/10.5524/102001_103000/102423/supplementary_tables/table_S3_splicing_pathogenic_curation.tsv). This new curation effort is composed of 161 variants covering a diverse range of disease phenotypes, with most diseases represented by fewer than 3 variants (Figure A.2A). A great number of these variants are not yet reported in ClinVar (N=90), and of those that are reported, a few (N=11) are incorrectly classified as VUS, with review status ranging from 0-1 stars (Figure A.2B). As further evidence of their pathogenicity, the variants are very rare in the general population, as most of them are absent from gnomAD [12], a widely used catalog of genetic variation across human populations (Figure A.2C).

The results showed that SpliceAI-derived methods outperformed the remaining tools. PDIVAS displayed the highest area under the ROC (auROC), followed by Pangolin, ConSpliceML and SpliceAI (Figure 4.3A). However, evaluation using single thresholds revealed lower performance than using auROC, which is based on multiple thresholds (Figure A.2D). As practical clinical applications usually require a binary decision, this prompted us to optimize reference thresholds for detecting splice-affecting pathogenic intronic variation outside of the canonical splice site regions (see VETA threshold recalibration). After recalibration, we reveal SpliceAI and Pangolin as the best tools (weighted normalized MCC > 0.92) to identify pathogenic variants using a single cutoff value (Figure 4.3B). As a practical outcome of this analysis, we provide recalibrated thresholds for different trade-offs between precision and recall (Table A.1).

4.5.1 Pseudoexon activation vs Partial intron retention

When available, we recorded information on the molecular consequences of each variant on splicing. Pseudoexon activation was the most frequent consequence of deep intronic variants (194 out of 242 in our dataset). We also identified 37 variants leading to partial intron retention due

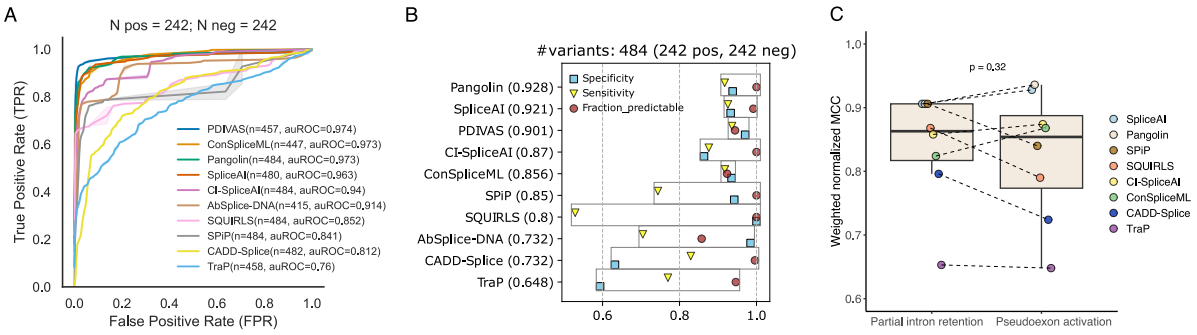

Figure 4.3: Pathogenic variant prediction of deep intronic variants affecting RNA splicing (81 variants from Vaz-Drago *et al.* [59] and 161 curated for this study). **A** - Receiving Operating Characteristic (ROC) analysis for all splicing-associated methods. **B** - Performance using optimized thresholds for intronic variation outside of canonical splice regions. The weighted normalized MCC was used to rank the tools. **C** - Performance using optimized thresholds on two different subsets of variants: variants leading to partial intron retention and variants leading to pseudoexon activation. PDIVAS and AbSplice-DNA were excluded as they do not score the two groups equally (PDIVAS only predicts variants located 50bp beyond the nearest splice site, while AbSplice-DNA scores variants within 100bp of any splice junction observed in GTEx data). Wilcoxon Signed Rank test, performed as a one-sided test, was used to compare the values between the two groups of variants.

to the usage of an alternative splice site. Exon skipping was observed in only 6 cases, consistent with previous observations that functional deep intronic variants are less commonly linked to this mechanism [192]. We next compared the tools' ability to detect pseudoexon activation and partial intron retention variants using the optimized thresholds. We hypothesized that the tools would perform better on the partial intron retention group since these variants are located closer to the splice sites than those that activate pseudoexons (Figure A.2E). Nonetheless, we observed no statistically significant differences between the two groups, with SpliceAI-derived methods performing slightly better in the pseudoexon activation group (Figure 4.3C).

4.6 Variable performance in variants associated with different molecular mechanisms

To gain further insight into the molecular mechanisms driving the splicing alterations, we generated datasets of alternative splicing events triggered by intronic variants occurring at different regions that are important for splicing regulation (Table 4.2, full datasets available at https://s3.ap-northeast-1.wasabisys.com/gigadb-datasets/live/pub/10.5524/102001_103000/102423/supplementary_tables/table_S6_splicing_per_region_datasets.tsv). We defined six categories (Figure 4.4, see Appendix A.1 for details). Within each region, we separately evaluated variants that trigger partial intron retention (via alternative splice site usage at annotated exons) and variants that lead to pseudoexon activation. Importantly,

contrary to the analyses performed above, we evaluate performance not based on the ability to distinguish between pathogenic and non-pathogenic variants but rather between variants that do (positive class) or do not (negative class) affect the mechanistic splicing outcome. This is relevant as a variant may, e.g., create a pseudoexon, thus affecting the outcome of splicing, without necessarily leading to disease. This decision was made as information on variant pathogenicity was not always available. We also switch to reporting performance using the area under the PR curve (auPRC), which is a metric to summarize precision-recall curve analysis. This decision was motivated by two factors. Firstly, some categories have unbalanced data, with fewer positive instances compared to negatives. The auPRC score provides a more nuanced evaluation of tool performance by focusing on the accurate identification of positive instances. Furthermore, it eliminates the need for a single universal cutoff, accommodating the fact that different categories may have distinct optimal thresholds.

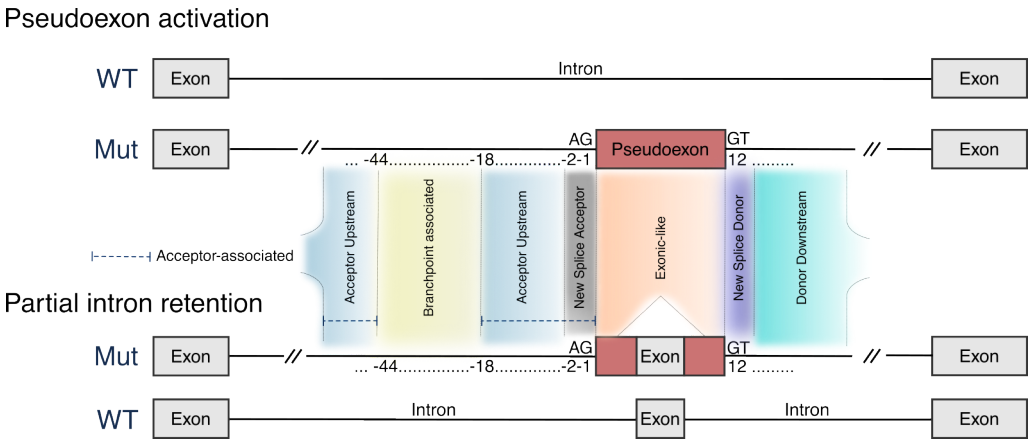

Figure 4.4: Schematic representation of the regions used to define each dataset. The figure displays the expected wildtype (WT) structure and the abnormal structure caused by the variant (Mut) for each of the two major groups: pseudoexon activation and partial intron retention. The red blocks indicate regions of the mRNA that are incorrectly spliced. Exceptionally, some branchpoint-associated variants result in exon skipping, which is not graphically represented in the figure.

4.6.1 Branchpoint associated variants

Branchpoint associated variants were defined as located 18 to 44 bp upstream of the splice acceptor of a cryptic or canonical splice site, either leading to pseudoexon activation, partial/full intron retention, or exon skipping. In addition, we confirmed that the variant either disrupted or created any of the four (increasingly relaxed) BP motifs described in [131]: YTNAY, YTNA, TNA, YNA. Particularly, we excluded any splicing-altering variants located 1 bp upstream of the branchpoint adenine. The final positive branchpoint-associated dataset (N=82) spans 7 different sources, with Leman *et al.* [151] contributing the most (N=31, Table 4.2). The negative variants

Table 4.2: Sources of data used to build region-specific splicing datasets.

Study	Variants*	Per category**	Description
Splicing-altering			
Vaz-Drago *et al.* [59]	81	AU=3;NSA=12;EL=10; NSD=39;DD=17	Manual curation of disease-causing deep intronic variants with experimental validations
Our curation	140	BP=7;AU=11;NSA=26; EL=19;NSD=43;DD=34	Manual curation of disease-causing deep intronic variants with experimental validations
Keegan *et al.* [60]	143	BP=1;AU=12;NSA=21; EL=13;NSD=71;DD=25	Characterization of hundreds of mutation events driving cryptic splicing via pseudoexon activation
Petersen *et al.* [193]	10	BP=1;AU=1;EL=7 ;DD=1	Characterization of pseudoexons activated by deep intronic mutations that do not create or strengthen cryptic splice sites
Tubeuf *et al.* [152]	3	EL=3	Benchmark of user-friendly tools for predicting variants affecting splicing regulatory elements
Jung *et al.* [192]	231	BP=25;AU=42;NSA=3; EL=56;NSD=37;DD=68	Identification of intronic mis-splicing mutations from RNA-Seq using a read ratios approach
Moles-Fernández *et al.* [153]	15	BP=1;AU=2;NSA=2; EL=2;NSD=7;DD=1	Benchmark of using both SpliceAI and user-friendly tools to identify deep intronic variants that disrupt splicing
Leman *et al.* [151]	31	BP=31	Benchmark of bioinformatics tools to predict BP as well as the impact of splicing variants occurring in the BP area
Zhang *et al.* [131]	16	BP=16	Genome-wide analysis of human branchpoints and development of a tool to score BP-associated variants
Splicing-neutral			
Moles-Fernández *et al.* [153]	98	BP=2;AU=35;DD=61	Benchmark of using both SpliceAI and user-friendly tools to identify deep intronic variants that disrupt splicing
Vex-seq [194]	277***	BP=59;AU=52, EL=119;DD=47	Vex-seq, a MPRA to test the impact of 2059 variants in splicing across 110 alternative exons
MFASS [195]	109	BP=34; AU=17; DD=58	Multiplexed functional assay (MFASS) that assayed the splicing effect of 27,733 ExAC variants
gnomAD [12]	261	NSA=64; NSD=197	Common (and hypothetically benign) variants that create true splice site motifs

* Total number of variants used from original study. Since several variants were duplicated across studies, we kept unique occurrences given the order they appear in the table (top to bottom).
** Number of variants contributing to each category. BP = Branchpoint-associated; NSA = New splice acceptor; NSD = New splice donor; AU = Acceptor upstream; DD = Donor downstream; EL = Exonic-like. Note: we could not assign a category to all variants of our curation, hence the lower number as compared to the original dataset (N=161).
*** Exceptionally, 119 variants from this study are exonic.

are located 18 to 44 bp upstream of an annotated splice site and had been shown not to affect splicing using the minigene-based reporter assays Vex-seq [194] and MFASS [195](Table 4.2). Because BP variants activating pseudoexons were scarce (N=4) and the molecular consequences of BP-associated variants are not clear-cut (e.g., the same BP variant may lead to intron retention and to exon skipping), we analyzed all the variants affecting the BP motif together. For this analysis, we additionally included four branchpoint prediction tools: SVM-BPFinder [129], BPP [130], LaBranchoR [132] and BPHunter [131]. Moreover, we included IntSplice2 [185] since it predicts splicing-associated SNVs at intronic positions overlapping the branchpoint region.

Pangolin was the best-performing method for BP-associated variant prediction with an impressive auPRC score of 0.93 (Figure 4.5, Figure A.3A). This result suggests that the training of Pangolin on multi-species data potentially contributed to increased robustness in capturing

the complexity of the branchpoint code. Among the tools specifically designed to predict BPs, LabRanchoR and BPHunter were very competent, ranking 2nd and 4th, respectively, with auPRC scores of 0.877 and 0.87. Conversely, BPP and SVM-BPFinder displayed more modest results.

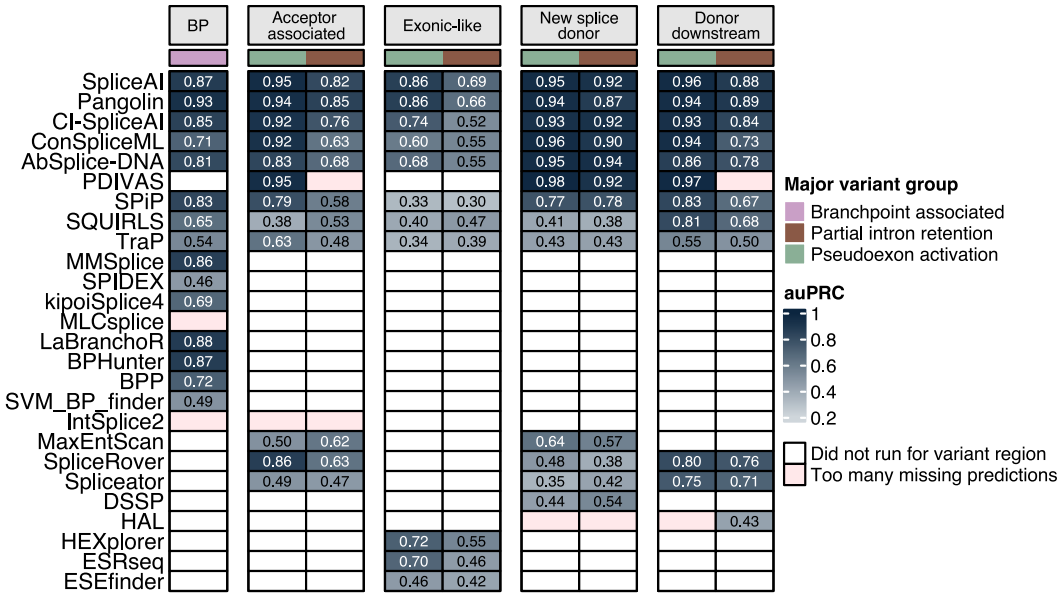

Figure 4.5: Global overview of the performance (quantified with auPRC) for all the datasets analyzed.

4.6.2 Acceptor Upstream and New Splice Acceptor variants

The *Acceptor Upstream* category refers to splicing-altering variants that mostly locate upstream (up to 18bp) of an existing cryptic splice acceptor and activate it. On the other hand, the *New Splice Acceptor* category contains variants that form new splice sites themselves (Figure 4.4). We collected negative variants differently for each of the two categories. For *Acceptor Upstream* variants, we extracted variants located upstream of annotated splicing acceptors that did not interfere with the splicing outcome, as demonstrated through MFASS, Vex-seq or Moles-Fernández *et al.* [153]. The BP region from 18 to 44 bp was excluded. Conversely, we assigned common (>5% allele frequency) deep intronic gnomAD variants that create new splice acceptor motifs as negative *New Splice Acceptor* variants (Table 4.2, see details in Appendix A.1). Despite creating a splice acceptor motif, these variants are considered non-functional due to their high prevalence in the general population. While it is theoretically possible that these variants do affect splicing (e.g, if they occur in non-essential genes where splicing alterations have little fitness effect), we confirmed that their genomic locations were not used as splice junctions in individuals from the GTEx [127] cohort.

The sets of splicing-altering variants we collected for each category were similar in size (71 and 64 variants for acceptor-upstream and new splice acceptor, respectively). However, when we split the variants according to the major molecular group (pseudoexon inclusion vs. partial intron retention), we obtained a very small number of new splice acceptor variants in the partial intron retention group (N=13), hence rendering their computational evaluation statistically limited. Therefore, for this particular analysis, we merged *Acceptor upstream* and *New Splice Acceptor* variants into a new *Acceptor associated* class so that we could have a reasonably large dataset to evaluate. As for the *Branchpoint associated* variants, we added IntSplice2 to the list of tools to evaluate. In addition, we included two splice site prediction methods that we customized to predict variant effects in VCF format: SpliceRover [123] and Spliceator [124].

SpliceAI, PDIVAS, Pangolin, ConSpliceML and CI-SpliceAI achieved good performance on pseudoexon-activating variants, with auPRC above 0.9 (Figure 4.5). However, when it comes to variants causing partial intron retention, performance drops considerably, with no tool achieving an auPRC score higher than 0.85 (Figure 4.5). Except for PDIVAS, which had a substantial amount of missing data for this analysis, the top tools remained unchanged, with Pangolin, SpliceAI and CI-SpliceAI displaying auPRC scores of 0.847, 0.816 and 0.765, respectively (Figure A.3C). Among the tools specifically added for this analysis, SpliceRover was the most competitive, ranking 6th in the pseudoexon group and 5th for partial intron retention variants (Figure A.3B,C).

4.6.3 Exonic-like variants

We consider here intronic variants that lie within either an activated pseudoexon or within an annotated exon that undergoes alternative splice site usage (Figure 4.4). We identified 110 splicing-altering variants to compare against 119 splicing-neutral exonic variants from Vex-seq (Table 4.2). After grouping the variants according to the major group, we obtained 78 pseudoexon-activating variants vs. 32 variants triggering partial intron retention. Accordingly, we randomly split the negative the variants between the two groups so that the final datasets were fairly balanced (84 and 35 variants for each group, respectively). For this comparison, we also included three approaches that quantify splicing regulatory elements that enhance or repress flanking splice sites: ESRseq scores [118], HEXplorer [164] and ESEfinder [163].

Once again, we observed better overall performance for the pseudoexon group compared to the partial intron retention group (Figure 4.5, Figure A.3D, E). Pangolin and SpliceAI were among the best tools in both major groups. Interestingly, HEXplorer and ESRseq performed better for the pseudoexon group than models that incorporate deep learning based predictions such as AbSplice-DNA or ConSpliceML (Figure A.3D).

Although SpliceAI performed best comparing to other methods, its pre-computed scores were configured to only report variant effects in a 50-bp window from the variant site. While this

window is fine for most variant types (the affected splice sites are usually close to the variant site), that may not be the case for pseudoexon-activating variants that could be located deep inside the pseudoexon (assuming a pseudoexon of the size of an annotated exon). Therefore, we selected the splicing-altering variants missed by SpliceAI using the optimized threshold of 0.05 (N=25) and used the SpliceAI Lookup API (https://spliceailookup.broadinstitute.org/, last accessed May 25th, 2023), to run the model using a larger maximum distance (500 bp). We observed that 9 out of 25 were correctly reclassified as splicing-altering, suggesting that SpliceAI performance may be underestimated when ignoring longer-range variant effects.

4.6.4 New Splice Donor variants

We identified 197 positive variants falling into this category (Table 4.2). For the negative set, we used variants that created a GT dinucleotide resulting in a splice donor consensus (GGTAAG), but that were unlikely to act as a cryptic splice site as they appeared in gnomAD with a population frequency >5% and were not observed to be used as a splice junction in GTEx individuals. We added SpliceRover, DSSP and Spliceator tools to the evaluation.

PDIVAS demonstrated the best performance in the pseudoexon activation group, achieving an auPRC score of 0.981. On the other hand, AbSplice-DNA outperformed other tools for partial intron retention variants with a performance metric of 0.94 (Figure 4.5, Figure A.3F, G). Similarly, SpliceAI, ConSpliceML, Pangolin and CI-SpliceAI exhibited excellent results, indicating that these models are very well-suited for predicting this category of variants. Importantly, we noticed a large performance gap between SpliceAI-related tools (plus SPiP) and the rest, which performed rather poorly (almost all tools with auPRC scores below 0.6, Figure 4.5). Considering that splicing-negative variants in this dataset create hypothetical splice donor decoys, we wondered whether tools that incorporate cryptic splice site scoring features using short sequence windows surrounding the variant site (Position Specific Scoring Matrix (PSSM)-based for TraP, information content-based for SQUIRLS) would predict negative variants as splicing-altering. Indeed, we observed a large proportion of false positives for these tools in the pseudoexon-activation group when using a single reference threshold for evaluation (1.0 for TraP and 0.98 for SQUIRLS). Conversely, deep learning based methods such as SpliceRover, DSSP and Spliceator may rely too much on the near-splice site features (despite using larger sequence contexts), hence the poor performance observed.

4.6.5 Donor Downstream variants

This category refers to all splicing-altering intronic variants located downstream of the cryptic splice donor event (N=146). Negative variants (N=166) are located downstream of annotated exons and were shown experimentally to have no impact on splicing outcomes (Table 4.2). As before, we included SpliceRover and Spliceator in the analysis. DSSP was excluded since it

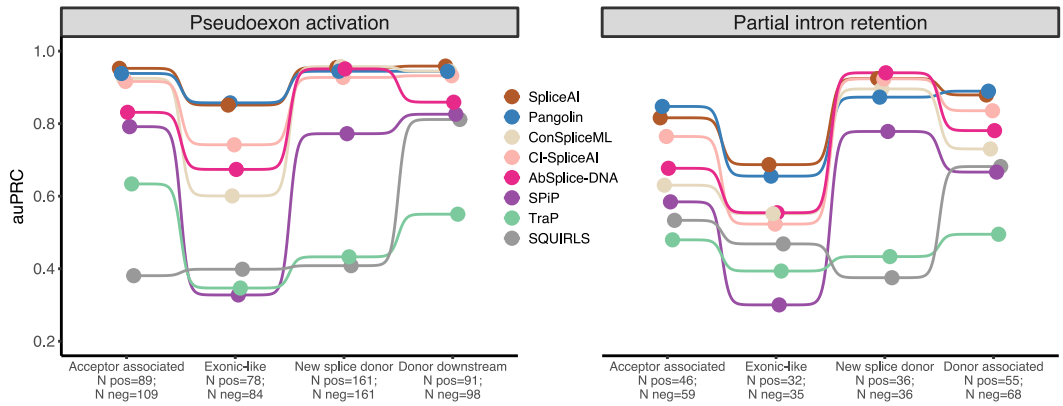

Figure 4.6: auPRC scores for the tools capable of predicting entire introns.

predicts splice sites at fixed positions in the input, but in this category, variant positions with respect to the cryptic splice donor are variable.

PDIVAS and SpliceAI excelled on the subset of variants triggering pseudoexon activation with auPRC scores of 0.969 and 0.959, followed by ConSpliceML, Pangolin and CI-SpliceAI, all with performance values above 0.9 (Figure 4.5, Figure A.3H). Regarding the partial intron retention subset, Pangolin and SpliceAI performed the best (auPRC scores of 0.89 and 0.879), with a larger difference for the tool ranked third, CI-SpliceAI (auPRC=0.836, Figure A.3I). Again, these results demonstrate the superiority of SpliceAI-derived approaches versus standard methods that engineer domain-specific features to score intronic splicing variation.

4.6.6 All regions combined

Next, we combined all the datasets to inspect the global performance of each major variant group. Eight methods were able to score all types of splicing variants in any intronic region. These tools were SpliceAI, Pangolin, ConSpliceML, CI- SpliceAI, AbSplice-DNA, SPiP, TraP, and SQUIRLS (Figure 4.6). Except for AbSplice-DNA, which scores intronic variants located up to 100bp away from splice junctions used in any GTEx tissue, all the methods were designed to score any given position in introns.

SpliceAI and Pangolin consistently ranked highly for all datasets (Figure 4.6). CI-SpliceAI, AbSplice-DNA and ConSpliceML were fair alternatives, especially for variants that create new splice donors. SPiP was particularly inadequate for exonic-like variants, but was the best non-deep learning-based method for the remaining categories (Figure 4.6).

Overall, we observed a trend of pseudoexon-activating variants being predicted more accurately than partial intron retention variants (Figure 4.6, Figure A.4A). However, when evaluating each tool individually, this trend did not reach statistical significance for the majority of them (Figure A.4B). A table with all performance

metrics for each tool and dataset analyzed in this section can be found at https://s3.ap-northeast-1.wasabisys.com/gigadb-datasets/live/pub/10.5524/102001_103000/102423/supplementary_tables/table_S7_splicing_per_region_datasets_results.tsv.

4.7 Assessing interpretability

We were interested in the extent to which these state-of-the-art tools give additional information to the user, besides the prediction. Among the tools that predict across whole introns, SQUIRLS and SPiP are the only ones intentionally designed to provide some interpretation of the outcome. SQUIRLS can generate HTML reports with short descriptions of why the model predicts pathogenicity and displays the contribution of each feature to the outcome. In addition, it draws figures to show the variant effect in the sequence context surrounding the variant. SPiP provides short interpretation tags describing the molecular consequences of the variants along with confidence intervals for the probability that the variant impacts splicing. Recently, a novel strategy was introduced to aid in the interpretation of splicing-associated variants, leveraging RNA-Seq data from more than 300,000 individuals [196]. This approach, SpliceVault, focuses on quantifying the relative prevalence of stochastic and unannotated splicing events in population-based RNA-seq data, enabling the prediction of the nature of mis-splicing induced by a variant. Given its innovative approach and the ability to provide interpretations for variant consequences, we included SpliceVault in our assessment.

We devised a procedure to evaluate how accurate the interpretations are against the biological ground truth (see details in Appendix A.5). We used the splicing-associated deep intronic pathogenic dataset analyzed before (Figure 4.3) and specifically selected variants with complete annotations, including molecular effect and functional consequence (N=221) for assessing interpretation quality. SPiP and SQUIRLS correctly predicted 170 and 121 variants, respectively, and those were selected for downstream analysis. In contrast, SpliceVault does not predict variant effects directly. Instead, it checks the mis-splicing occurring in the surroundings of an annotated exon of interest. As a result, we did not include pseudoexon-activating variants because SpliceVault cannot provide information about such an outcome (despite potentially identifying one of the two splice junctions of the pseudoexon). This left us with 37 variants for analysis (available at https://s3.ap-northeast-1.wasabisys.com/gigadb-datasets/live/pub/10.5524/102001_103000/102423/supplementary_tables/table_S9_SpliceVault_presults.tsv). Our evaluation revealed that SQUIRLS, SPiP, and SpliceVault were able to provide correct interpretations (within the limitations of each approach) for a considerable fraction of the variants. However, for many others, no interpretation could be found. Specifically, SPiP lacked interpretations for 46 variants, SpliceVault for 22 variants, and SQUIRLS for 21 variants (Figure 4.7A). In the case of SpliceVault, this accounted for more than half of the analyzed

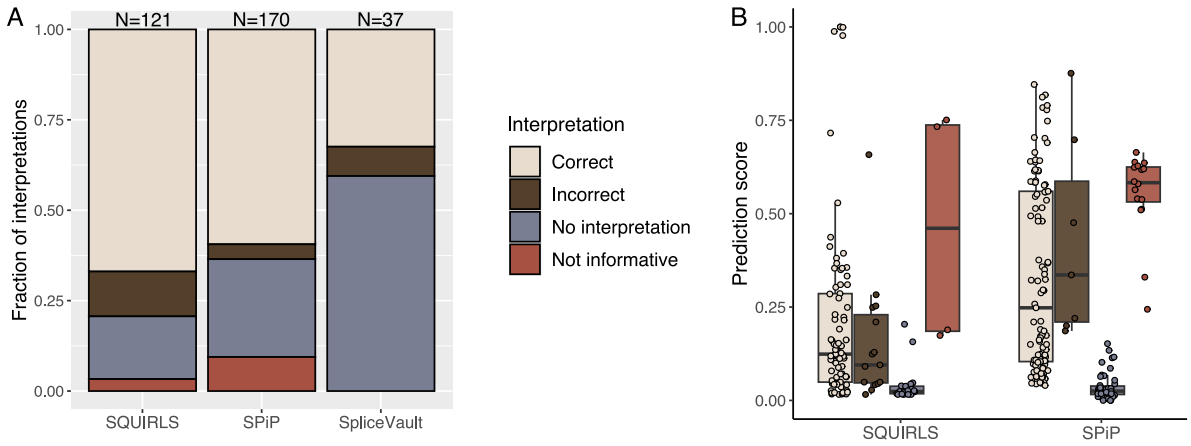

Figure 4.7: Tool's interpretability assessment. **A** - Assessing the quality of the interpretations for SQUIRLS, SPiP and SpliceVault. Within each bar, the height of each category represents the fraction of variants assigned to the given interpretation quality tag. The numbers above the bars indicate how many pathogenic variants were used. **B** - Distribution of SPiP and SQUIRLS prediction values for each of the interpretation categories.

variants (22 out of 37). Further inspection of these results showed that most of these variants create a core splice site dinucleotide, and this type of mis-splicing event is not appropriate to be captured by SpliceVault. Regarding SQUIRLS and SPiP, and looking at the prediction score distribution for each category, we observed that the variants with no interpretation have the lowest scores (Figure 4.7B). On the other hand, correct explanations are spread across the full score range. Interestingly, SPiP explanations that are not informative (events with no association with any splicing mechanism) have the highest median score range, showing that strong effects are not necessarily easier to explain.

4.8 Predicting splicing changes across tissues

Of the tools evaluated in this study, Pangolin and AbSplice-DNA can both predict splicing outcomes in a tissue-specific fashion. We decided to use AbSplice-DNA alone for this analysis. Pangolin was trained on sequences and splice site usage levels from four tissues across four species (human, rhesus macaque, mouse and rat). However, the default settings of Pangolin are tissue-agnostic and it requires additional customizations to get tissue-specific variant effect predictions. On the other hand, AbSplice-DNA provides pre-computed tissue-specific predictions. Moreover, it combines tissue-specific splicing annotations created from GTEx data with DNA-based prediction models, enabling it to predict variant effects in more tissues (49). We aimed to evaluate whether AbSplice-DNA predictions of disease-causing variants are enriched for the tissues that are most strongly affected by the disease.

Using the splicing-associated variant dataset described above (N=242, Figure 4.3), we

determined, when possible, the GTEx tissue most closely associated with the given disease based on the Human Phenotype Ontology (HPO) [197] (see Appendix A.6 for details). We selected the 155 variants that AbSplice-DNA predicted correctly (see full dataset at https://s3.ap-northeast-1.wasabisys.com/gigadb-datasets/live/pub/10.5524/102001_103000/102423/supplementary_tables/table_S10_AbSplice_DNA_tissue_assignments.tsv), which excluded all the variants causing two of the most common diseases in our dataset: Becker muscular dystrophy and Duchenne muscular dystrophy. In addition, 35 variants were not evaluated since they were not assigned to any particular tissue (e.g. systemic diseases, or diseases affecting tissues not represented in GTEx, such as the retina), leaving 120 variants to analyze. Considering disease variants associated with only one GTEx tissue, we observed enrichment of the expected tissues to a limited extent (Figure 4.8, Figure A.5A). For example, Hypertrophic Cardiomyopathy variants were highly enriched in the heart tissue, an Ataxia with Oculimotor Apraxia variant was predicted to affect the cerebellum and a Congenital hypothyroidism variant was enriched for thyroid. Interestingly, variants associated with blood disorders (Factor VII deficiency and Afibrinogenemia) have the highest prediction scores in the liver, which is not surprising, since the liver plays a crucial role in the production of clotting factors, including factor VII and fibrinogen (Figure 4.8). However, other tissue-specific predictions had unclear interpretations, such as the enrichment of testis for several diseases, the brain cerebellum in Adenomatous Polyposis (associated with colon and rectum, Figure 4.8B), or the skeletal muscle in Fabry disease (primarily linked to other tissues such as heart and kidneys, Figure A.5A). In addition, 40 variants displayed the same score across all tissues, which does not reflect the expected biology, especially for some diseases associated with a single tissue (Figure A.5B).

4.9 Discussion

We used two different datasets to study intronic variants causing human disease. ClinVar is a database that has been widely used for this purpose. Nevertheless, to the best of our knowledge, it has not been used to evaluate performance as a function of distance to the splice sites. Averaging performance across all bins, we found that splicing-associated tools performed the best overall on ClinVar data. Importantly, we observed a decrease in performance immediately after the two splice site positions, with a particularly noticeable decline at a distance of 11 base pairs from the closest splice site. These results demonstrate the extent to which these methods are biased to predict splice site variants, whereas smaller effect-size variants deeper inside the intron go mostly unnoticed. For many of the tools, such as S-CAP or MLCsplice, this is not unexpected, as they were not designed to predict variants in deep intronic regions. In addition, we observed that some of the variants that appear deep-intronic in the clinically-relevant transcript are exonic or located close to the splice sites in other isoforms of the associated gene. Therefore,

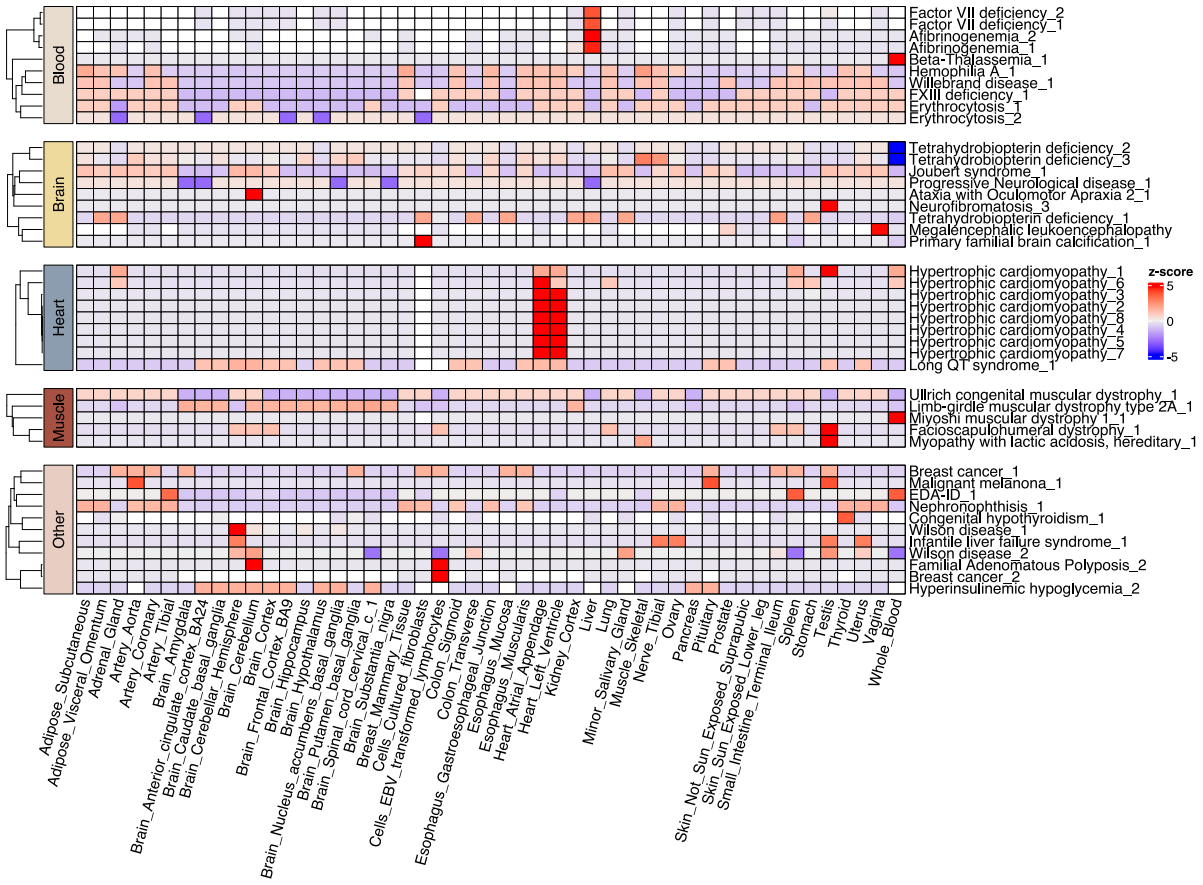

Figure 4.8: Tissue-specific predictions made by AbSplice-DNA for a set of disease-causing variants associated with a single tissue, according to HPO. Phenotype names are displayed in rows, GTEx tissues predicted by AbSplice-DNA are in columns. High z-scores represent tissues for which the variant effect is stronger compared to other tissues.

and according to the American College of Medical Genetics and Genomics and the Association for Molecular Pathology (ACMG-AMP) guidelines [198], we recommend considering multiple isoforms when interpreting deep intronic variants, especially when the canonical isoform is not highly expressed in the tissue of interest [199].

Additionally, we curated a diverse set of pathogenic deep intronic mutations that exclusively affect splicing. Tools that predict across all intronic regions, notably SpliceAI-derived models, showed satisfactory performance. Many variants in this dataset generate new splice sites deep within introns, activating pseudoexons. We speculate that sequence-based models that predict splice sites are particularly well suited to predicting this class of variants, likely because the pseudoexons resemble the sequence context of authentic exons [193] that were presented during their training.

To better understand performance differences between classes of variants, we collected a diverse set of experimentally tested splicing-associated variants, and evaluated the tools' ability to distinguish them from similar non-splice-altering variants. Region-specific analysis revealed substantial differences in performance. In agreement with previous studies [153, 200], we found that variants affecting putative exonic splicing regulatory elements were among the hardest to predict. The binding motifs of many splicing factors are highly degenerate or even unknown, and their impact on splicing largely depends on the cell type [201, 202]. Nevertheless, such complexity appears to be better captured by SpliceAI and Pangolin than by tools with built-in domain knowledge.

The recent progress achieved through deep learning models that work as black boxes has raised concerns about their deployment in sensitive domains such as healthcare [203]. Because practitioners are interested in understanding how these AI systems make decisions, we assessed the capacity of these models to provide interpretable outputs when predicting disease-causing variation associated with splicing defects. Although most sequence-based models, such as SpliceAI, provide some information beyond the prediction score, namely the distance of the variant to the affected mRNA position, it is only possible to obtain insight into the inner workings of the model by applying external explainability techniques [30]. On the other hand, SQUIRLS and SPiP are intrinsically more interpretable by design. The models were frequently able to correctly identify the type of splicing alteration. However, these models suffer from an accuracy-interpretability trade-off since the performance across evaluations was lower than that of black box models. The recently published SpiceVault portal (`https://kidsneuro.shinyapps.io/splicevault`) also provides an accurate interpretation of the nature of mis-splicing defects, however, it does have limitations that are specially pronounced when dealing with variants deep in the introns. In particular, it cannot properly analyze pseudoexon activation events or cryptic splicing caused by variants that create new splice sites at the core dinucleotide motif. Note that to our knowledge, no tool exists that can provide higher-order mechanistic interpretations, such as identifying the particular splicing factors or regulatory motifs involved.

Another promising research avenue is the prediction of splicing abnormalities in a tissue of interest, which AbSplice-DNA offers. The model could accurately detect some tissue-specific differences relevant to human disease, yet it was unreliable for the majority of variants. Nonetheless, we acknowledge that the introduction of SpliceMaps [142], which provides information on splice site usage across GTEx tissues, combined with RNA-Sequencing of clinically accessible tissues (CATs), is expected to enhance the prediction of functional intronic variants [142], particularly in diseases where the splicing landscape of the relevant non-accessible tissue is appropriately represented by one of the CATs [204].

4.9.1 Practical recommendations

We advocate using deep learning based solutions to obtain maximally accurate predictions. SpliceAI and Pangolin consistently ranked high for intronic variants associated with splicing, both for the prediction of pathogenicity and of altered splicing. We determined optimal thresholds for deep intronic regions (SpliceAI=0.05, Pangolin=0.053) for clinical purposes. However, it is important to note that despite the diversity of genes, phenotypes and molecular mechanisms covered in our dataset, users should be mindful that optimal thresholds can vary depending on the variant class or affected exon [205].

SpliceAI and Pangolin are usually run programmatically on the command line. However, if usability is a primary concern and users have a limited number of predictions to make, the Broad Institute offers a convenient web application. The web application (https://spliceailookup.broadinstitute.org/) incorporates both SpliceAI and Pangolin. For SpliceAI, it not only provides the conventional delta score (mutated - reference) but also presents the raw splice site probability predicted by the model. This can be particularly useful for certain situations. For instance, when a splice site is already predicted with a high score in the reference sequence (e.g., 0.85), the delta score for a splice-promoting mutation can only be low (no more than 0.15, in this example). This is because SpliceAI scores are capped at 1. This context is important for the correct interpretation of the delta scores. With this in mind, it is also worth considering SpliceAI-visual [206], available at https://mobidetails.iurc.montp.inserm.fr/MD. SpliceAI-visual handles complex variant types, and employs raw SpliceAI scores to generate graphical outputs that are easier to interpret. If the number of variants makes it unfeasible to use these web applications, but the user does not have the computational know-how to work on the command line, CI-SpliceAI is a good alternative, since its online service (https://ci-spliceai.com/) allows the input of multiple variants in a VCF-like format. Practitioners, however, may suspect that splicing is not the mechanism disrupted by a particular mutation. In this scenario, we recommend using CAPICE since it was the best whole genome predictor on ClinVar data, although with very limited performance.

Region-specific splicing benchmarks revealed additional insights for tool usage. We recommend using Pangolin to prioritize variants in branchpoint regions (-18 to -44 bp upstream of splice acceptor). LabRanchoR and BPHunter were the best branchpoint-specific tools in our evaluation and can also be considered. SpliceAI and Pangolin were the most effective at scoring acceptor-associated variants (splice acceptor creating or polypyrimidine tract variants upstream of cryptic splice acceptors). Including other sequence-based deep learning models that use smaller sequence contexts did not provide additional value. Intronic variants affecting splicing regulatory elements within cryptic exons are hard to predict. We endorse using SpliceAI with larger windows surrounding the variant site (setting the distance parameter to the maximum). In addition, classical approaches such as HEXplorer might come in handy for

specific cases, such as assessing the potential impact of a variant on exon-defining regulatory motifs. Finally, SpliceAI-inspired models (Pangolin, CI-SpliceAI) and models that incorporate SpliceAI predictions as features (ConSpliceML, AbSplice-DNA, PDIVAS) can effectively predict new splice donor and donor-downstream variants. However, to keep the number of different tools to use to a minimum, we suggest using the original SpliceAI model.

Nonetheless, it is noteworthy to mention the impacts of using pre-computed scores as the strategy for variant prioritization. The current version of SpliceAI pre-computed scores (v1.3.1) does not include predictions for insertions and deletions larger than 1 and 4 nucleotides, respectively. In addition, the limit of 50 base pairs as the distance around the variant site to extract variant effects prevents SpliceAI from identifying other variant classes, such as exon skipping, when the variant exerts its effects at more than 50 base pairs from the affected exon.

Finally, when interpretable outcomes are important the choice of the strategy may depend on the use case. There is currently no option covering all possible mis-splicing scenarios, and each method assesses interpretability differently. SpliceVault is a recently published web application that is effective when interpreting intronic variants leading to exon (or multi-exon) skipping, or partial intron retention through activation of pre-existing cryptic splice sites. Alternatively, SQUIRLS can be applied, as the software is well-designed, thoroughly documented and generates HTML reports that practitioners can intuitively inspect. Nonetheless, it does not handle pseudoexon activation consequences (for that, SPiP is recommended). In addition, it should not be solely relied upon as a prediction tool, as it is not as performant as other models.

4.9.2 Final remarks

We comprehensively assessed functional intronic variation occurring far from annotated splice sites. As a result, we make available to the community region-specific datasets that can be used to evaluate new models on variants whose molecular consequence is known. These datasets will assist developers in identifying potential limitations of the model and highlighting variant types that it is more prone to fail on. Additionally, we encourage developers to make their models publicly available by sharing them on open-source platforms to facilitate their reuse [82, 207].

Sequence-based models based on Convolution Neural Networks architectures are still the state-of-the-art approach for splicing variant prediction. However, artificial intelligence is rapidly evolving, and we have seen the emergence of Transformer-based architectures being applied to other variant effect prediction tasks, e.g., effects on gene expression [208] or on protein function using large protein language models [209]. As a result, increasingly complex models are expected to effectively tackle open questions in splicing regulation, such as better capturing the synergistic effects of splicing regulatory elements. However, the community must be aware of the possible implications these models bring, such as a lack of transparency and decreased ability to generate mechanistic hypotheses.

Chapter 5

Interpreting SpliceAI

In this chapter, we present a comprehensive set of experiments to inspect whether SpliceAI is sensitive to known biological properties, particularly the binding motifs of known RBPs. Given RBP's influence in regulating alternative splicing (Section 2.1.1), assessing SpliceAI's ability to use their binding motifs as predictive features would establish the model as a valuable resource for studying alternative splicing.

To achieve this, we first leveraged large-scale publicly available data to generate multiple datasets relevant to the task. Next, we developed a modular pipeline to perform model ablations at scale, enabling us to interrogate, explore, and visualize whether SpliceAI utilizes these motifs as features in its predictions. We summarize the contributions of this chapter as follows:

- Generation of paired datasets to study alternative splicing regulation by individual RBPs.
  DOI: https://zenodo.org/records/11193459
- Development of MutSplice, a pipeline to perform *in silico* sequence perturbations with the aim of interpreting SpliceAI.
  Code: https://github.com/PedroBarbosa/mutsplice
  Reproducibility: https://github.com/PedroBarbosa/mutsplice/tree/main/notebooks

5.1 Generation of RBP-specific datasets to study splicing regulation

We took advantage of public RNA-Seq data from the ENCODE consortium [113] to identify exons whose inclusion levels (PSI) change when an RBP known to regulate alternative splicing is knocked down (Figure 5.1A). These exons, referred to as *knockdown-sensitive exons*, are more likely to be directly or indirectly regulated by such RBPs, thus providing insights into their regulatory mechanisms.

5.1.1 Differential splicing analysis

We used ENCODE RNA-Seq data generated from Short Hairpin RNA (shRNA) knockdown experiments targeting individual RBPs. For each RBP, there were two knockdown and two control (samples with no RBP target) replicates. Because in the original paper [113] authors used an older version of the human genome (hg19) along with old genome annotations (GENCODE v19), we reanalyzed ENCODE data aligned to the hg38 genome build. We started from RNA-Seq genome alignment files for RBPs associated with splicing regulation (according to [113]). There were 72 knockdown experiments available for the HepG2 cell line, which were used for downstream analysis (details on accession IDs in Table B.1).

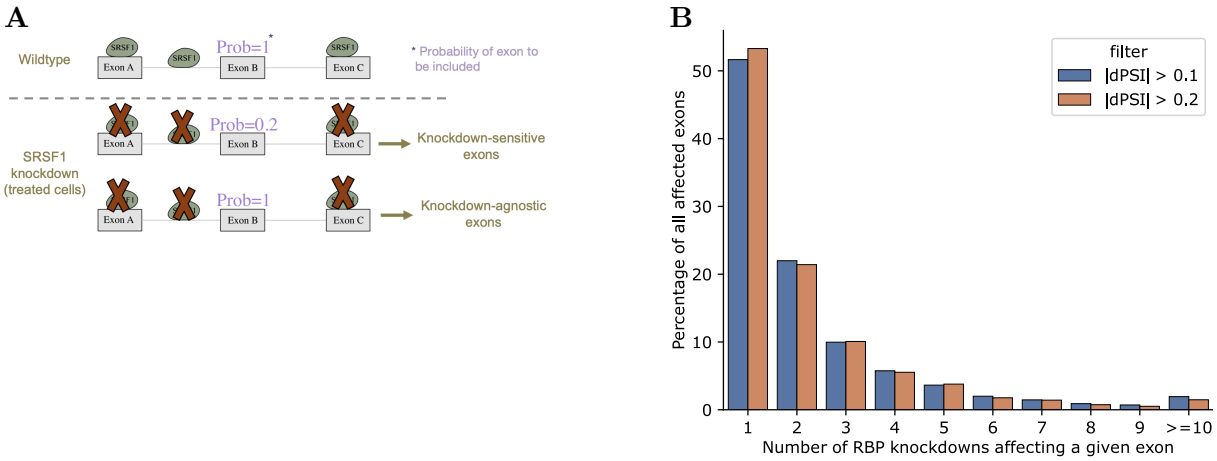

Figure 5.1: Splicing analysis of RNA-Seq involving the knockdown of individual RBPs. **A** - Illustrative example of the knockdown of SRSF1, an RBP known to regulate alternative splicing. The knockdown of SRSF1 will affect the inclusion of exons that are directly or indirectly regulated by SRSF1 binding. The outcome of the analysis is the identification of exons that are either sensitive or agnostic to the knockdown of SRSF1. **B** - Percentage of all exons affected by at least one knockdown experiment (out of 72) shared by 1 or more experiments, using two widely used dPSI filtering thresholds.

We employed rMATS v4.1.2 [108] on each RBP knockdown experiment to detect differentially spliced events between the two knockdown replicates vs. the two control replicates. rMATS was run with GENCODE annotations v44 [210], and specifically tweaked with `--cstat 0.05`. Significant knockdown-sensitive events were identified with a $|dPSI| > 0.1$ using a False Discovery Rate cutoff of 0.05. Non-changing events, assumed as knockdown-agnostic controls, were defined as those exhibiting negligible $|dPSI|$ variation (< 0.025). We performed further analytical steps to ensure the high quality of the exon sets. First, we applied a read coverage filter by retaining events where the median coverage across replicates per condition for the isoform with more read counts was higher than 7. Then, we exclusively focused on exon skipping events in protein-coding genes and filtered out unannotated exons (pseudoexons) and the first or last exons of genes. In addition, we excluded duplicate exon-skipping events by picking the transcript

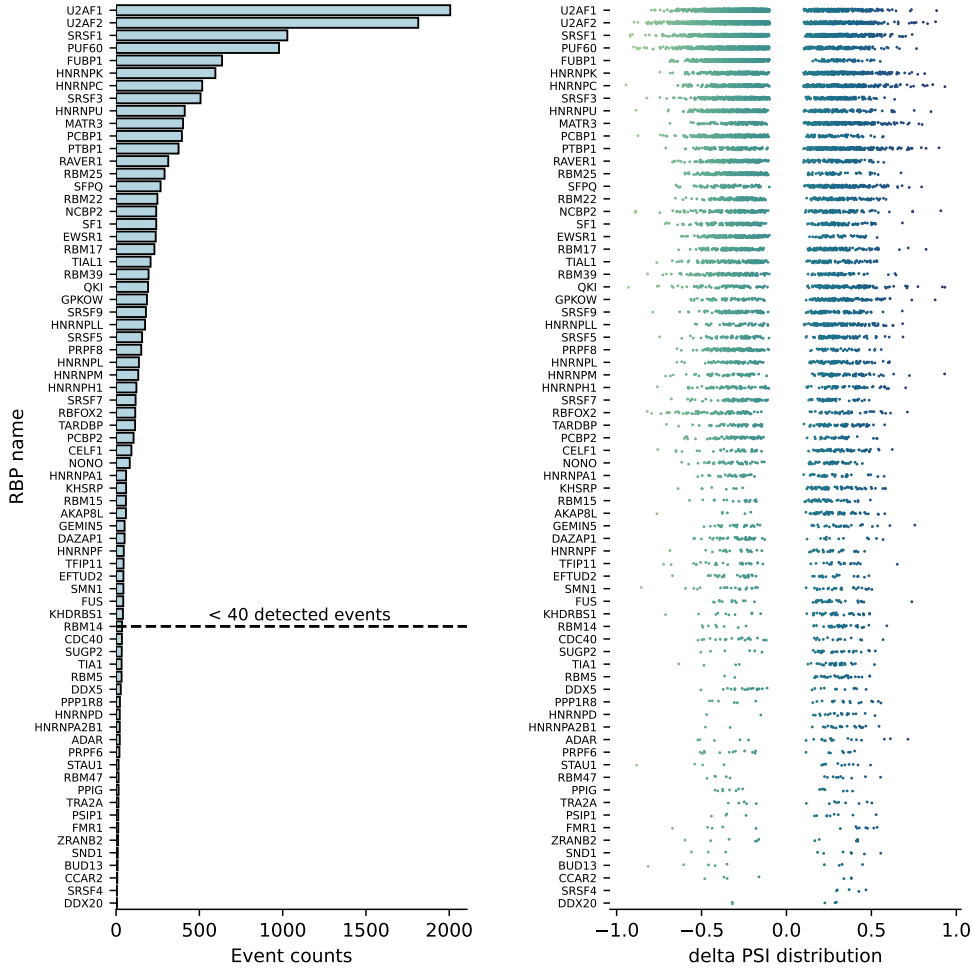

Figure 5.2: Summary of differential splicing analysis for each RBP knockdown involved in splicing regulation. Left: Number of events. Right: dPSI distribution. All downstream analysis were performed with RBP knockdown experiments with more than 40 differentially spliced exons detected.

with the highest biological importance (based on the presence of transcript flags such as MANE selected, CCDS, or APPRIS). Finally, we investigated whether the initial, somewhat arbitrary, choice of a $|dPSI|$ filter of 0.1 was appropriate for subsequent analyses, given that a threshold of 0.2 is also frequently used. Specifically, we assessed the number of exons displaying differential splicing in multiple RBP knockdown experiments. Regardless of filter stringency, over 50% of all differentially spliced exons were unique to a single RBP knockdown (Figure 5.1B). This high specificity is desirable for interpretability analysis, as we pose that exons sensitive to a single knockdown are easier to explain than those complex exons that are sensitive to multiple RBPs. Consequently, given the similar specificity between filters, we retained all exons with $|dPSI|$ > 0.1 as it preserved more data. This decision yielded 15,235 events detected across all RBP

knockdown experiments, covering 6,659 unique exons.

Unsurprisingly, the number of events detected per RBP knockdown experiment is highly heterogeneous, with important splicing factors such as U2AF1, U2AF2 and SRSF1 yielding the highest number of differentially spliced exons, with more than 1,000 events each (Figure 5.2, left panel). The effect sizes of each RBP knockdown are mostly moderate, with very few events displaying drastic changes in splicing ($|dPSI| > 0.9$, see Figure 5.2, right panel). The low number of events observed in several RBPs can be attributed to both biological and technical reasons. For instance, some RBPs have a more specific role in splicing regulation, hence affecting a smaller number of exons [211]. In addition, the number of detected events is directly influenced by the expression profile of the target RBP in the HepG2 cell line, as different cell types express RBPs at different levels. If an RBP is expressed at low levels, its knockdown is expected to have minimal effects on the cell. Moreover, it is worth acknowledging the importance of technical aspects of gene knockdown experiments, as off-target effects of the designed short hairpin RNAs [212] and their GC content [213] may influence the efficiency of the gene knockdown, and consequently affect the bioinformatics analysis outcome.

5.1.2 Paired datasets' generation

In addition to analyzing SpliceAI behavior using exons sensitive to RBP knockdowns (Figure 5.2), we aimed to extract control sequences - exons unchanged upon RBP knockdown - to pair with such knockdown-sensitive counterparts. To generate control pairs, we established a procedure to ensure that the control exons are as similar as possible to the knockdown-sensitive exons at gene architecture level. The rationale for this approach is grounded in the understanding that gene architecture features, such as exon/intron length or GC content, can influence splicing outcomes [214]. Therefore, we aimed to study whether SpliceAI uses features (sequence motifs) reflecting RBP binding, while controlling for potential confounding effects of other types of features. Our hypothesis posits that if SpliceAI has learned the biology grammar of RBP binding, we would observe more motifs associated with the specific RBP influencing SpliceAI predictions within the sequences of the knockdown group compared to their control pairs, regardless of gene architecture features.

Initially, we built a transcript-aware genome cache by summarizing the length and GC content of all exons (along with surrounding introns) in the human genome, based on GENCODE v44 annotations. Then, for each of the 72 RBP knockdown experiments, we gathered the knockdown-agnostic exons ($|dPSI| < 0.025$) and excluded those exhibiting alternative splicing in any of the other 71 knockdown experiments. Various strategies were then tested to select the closest control exon (Table 5.1), all of which involved assessing some sort of distance between a knockdown exon and all available control exons, ultimately selecting the control exon with the lowest distance. Thus, for each RBP knockdown experiment, we iterated over

Table 5.1: Strategies tested to extract the control exon that minimizes the feature vector distance the most to the target knockdown-sensitive exon.

Strategy tag	Features used [1]	Are features scaled? [3]	Metric	Median CV accuracy
euclidean_l	GC and Length [2]	Yes	Euclidean distance	0.501
cityblock_l	GC and Length	Yes	Manhattan distance	0.508
max_l	GC and Length	Yes	Max feature difference	0.510
euclidean	GC	No	Euclidean distance	0.534
cityblock	GC	No	Manhattan distance	0.543
max	GC	No	Max feature difference	0.533

[1] Feature values were used for three different genomic locations: The target exon, and the upstream and downstream introns. Hence, a total of six features were used when length-based features were included, and three when only GC content-based features were used.
[2] Given the large differences in intron lengths, raw values were transformed to log10 scale.
[3] When length-based features were used to calculate the distance, all features were standardized by removing the mean and scaling to unit variance.

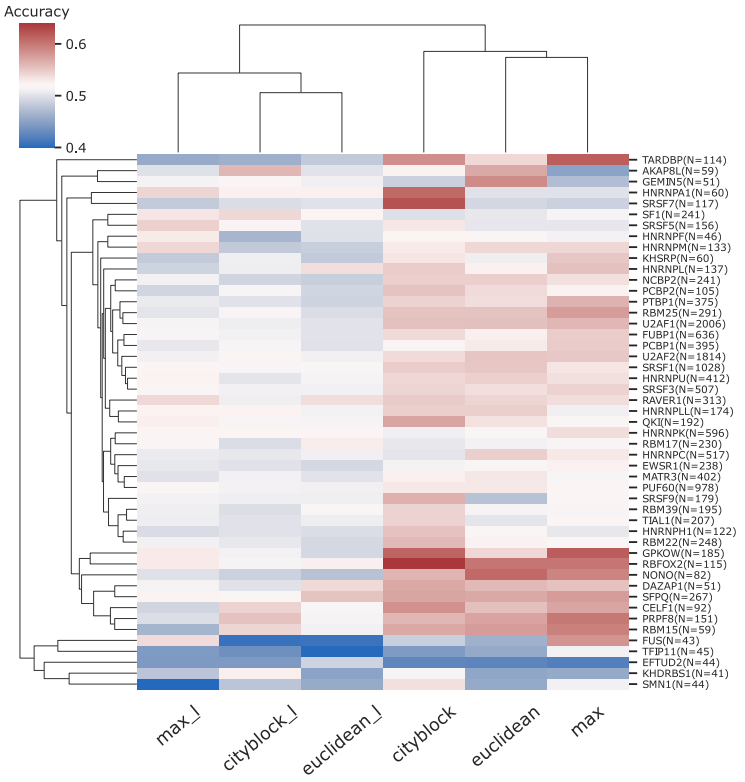

Figure 5.3: Heatmap displaying the accuracy scores of classifiers designed to distinguish exon groups across various paired datasets generated using diverse strategies. Each cell represents a decision tree trained on a specific strategy and RBP knockdown experiment. Decision trees were trained for RBP knockdown experiments with more than 40 differentially spliced exons detected (Figure 5.2, N=49), using scikit-learn v1.2.1 with parameters set to `random_state=0, min_samples_leaf=3, max_depth=5`. Performance was assessed by computing the mean accuracy of test sets from a stratified cross-validation procedure with 10 splits. The `N=` at each row label indicates the number of exons belonging to the knockdown class (Figure 5.2, they are the same across strategies). The full dataset size is twice that number, when considering the exons of the control class (which vary across strategies).

all knockdown-sensitive exons, selecting the closest control exon without replacement, ensuring that each control exon is used only once.

To select the best strategy for control exon extraction, we created a decision tree classifier for each paired dataset generated and tasked it to distinguish sequences from the knockdown-sensitive and control groups based on the genome architecture features alone. The idea is that the best strategy would be the one that harbors the worst cross-validation classification accuracy, which would indicate that the control exons are indistinguishable from the knockdown-sensitive exons based on gene architecture features, thereby aligning with our goal. We observed that incorporating exon/intron length features and using the Euclidean distance harbored the best results, with a median accuracy of 0.501 across all RBP knockdown experiments used (Table 5.1, Figure 5.3). Indeed, distribution of each feature is very similar between the knockdown-sensitive and control groups (Figure 5.4, Figures B.1 to B.5). Hence, we moved forward with assurance using these datasets for subsequent analysis.

5.1.3 SpliceAI predicts differently exons sensitive to RBP knockdown

We first asked how SpliceAI would predict the sequences of the extracted datasets. We reasoned that knockdown-sensitive exons, observed to undergo alternative splicing, would be predicted with lower probabilities than the control sequences. To test this hypothesis, we extracted the surrounding context of each exon (5000bp upstream of the splicing acceptor, 5000bp downstream of the splicing donor) and fed them to SpliceAI. We observed skewed distributions on the differences between exon groups (Figure 5.5A), which shows, on average, higher exon scores in the control group over the knockdown group. These differences were statistically significant in all paired datasets, except for DAZAP1 (Figure 5.5B). In addition, control exons were predominantly predicted with values close to 1 (Figure B.6), which is consistent with the notion that exons with strong splice sites are less susceptible to alternative splicing [43].

In summary, SpliceAI predicts alternatively spliced and constitutive exons differently. Considering that we controlled for the potential confounding effects of gene architecture features, these results suggest that SpliceAI has learned predictive sequence features of alternative splicing.

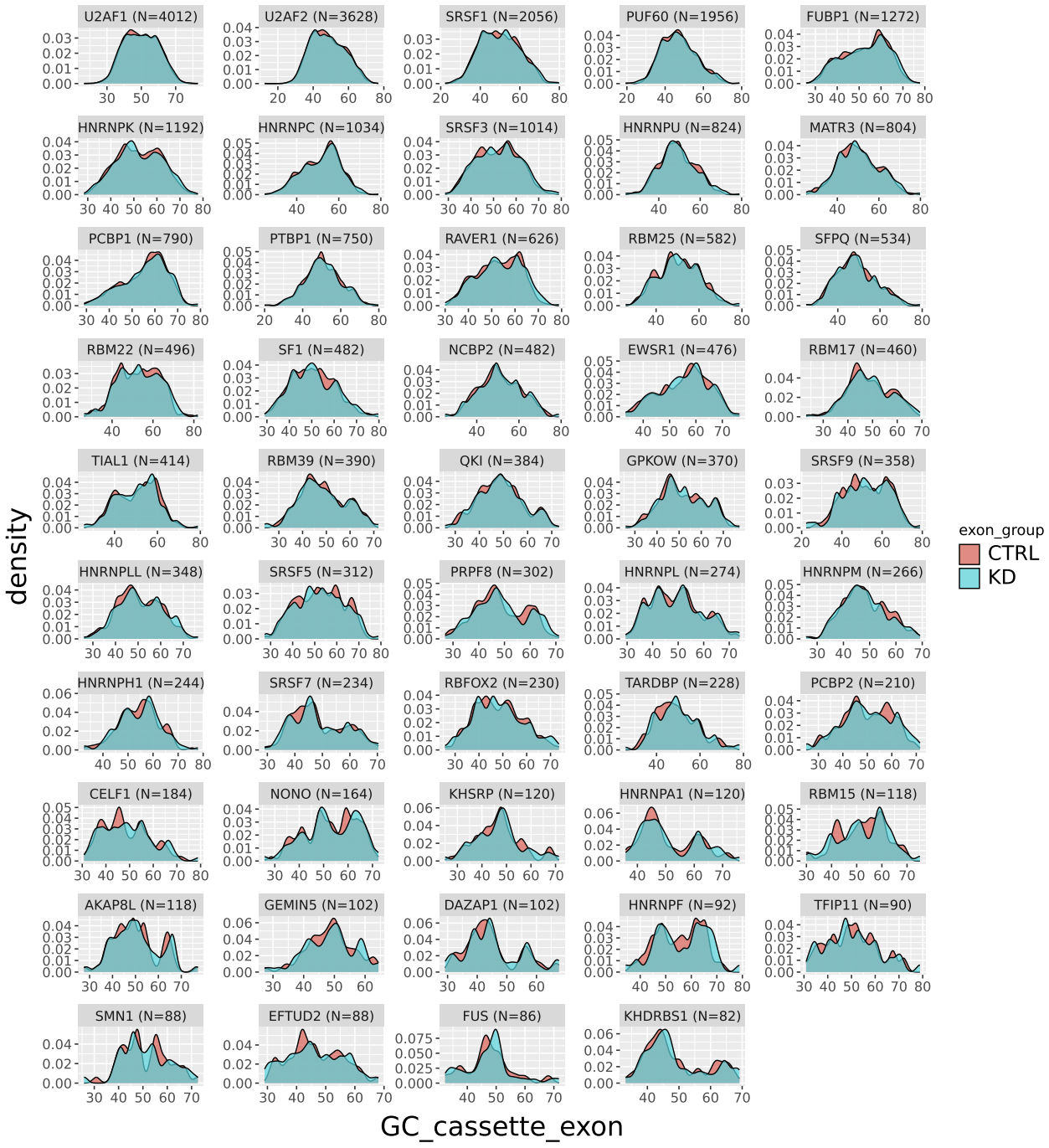

Figure 5.4: Distribution of GC content values for cassette exons in each paired dataset. The `N=` at each facet (a single RBP paired dataset) indicates the total dataset size, including knockdown-sensitive and control exons.

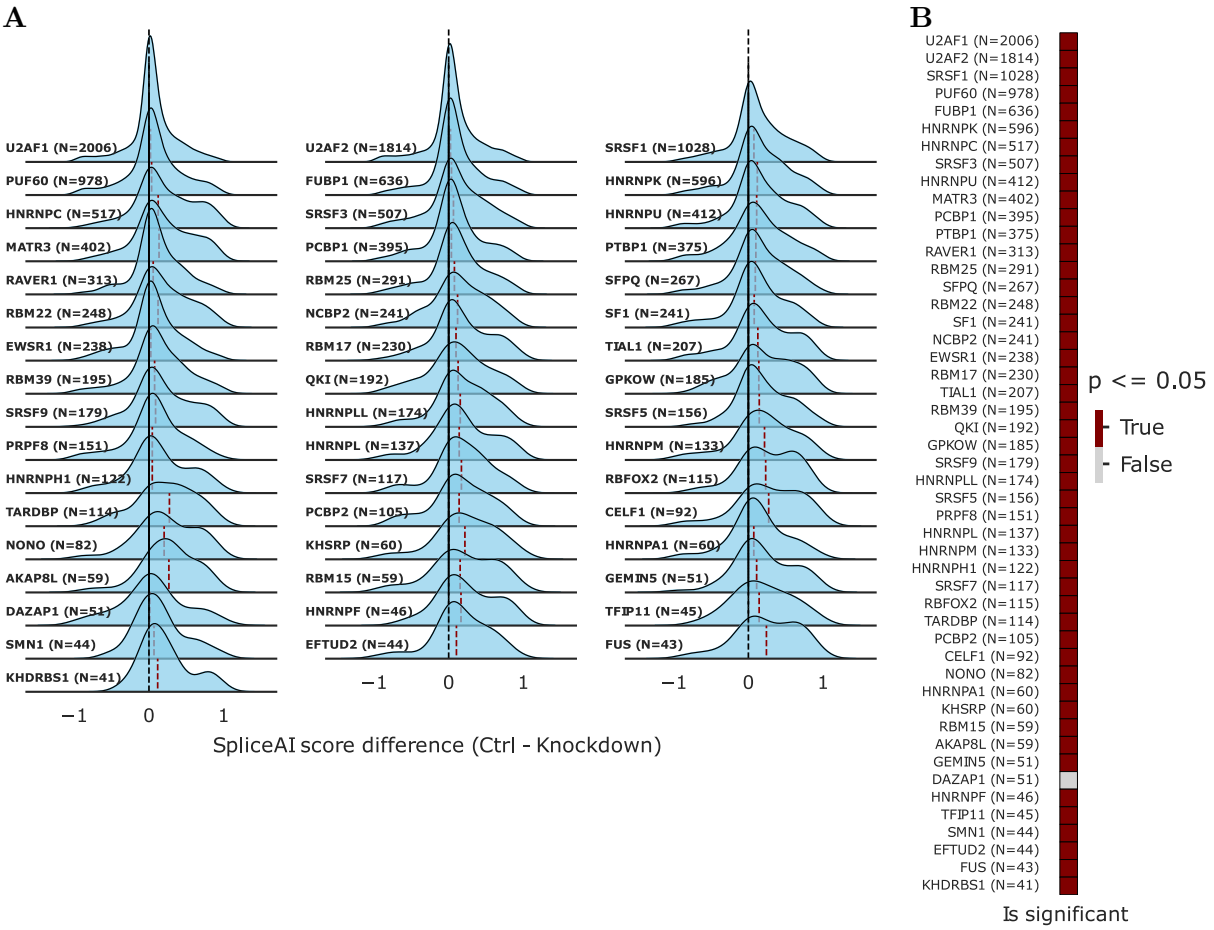

Figure 5.5: Quantifying differences in SpliceAI predictions across exon groups. **A** - Distribution of differences between exon pairs across 49 paired datasets. Each data point represents the subtraction of the control exon score from the knockdown exon score. The exon score is the mean of SpliceAI predictions at the acceptor and donor positions. Dashed vertical red lines indicate the median value of the distribution. **B** - Significance of the differences between exon groups assessed by paired t-tests. P-values were corrected for multiple testing using the Holm method. In both plots, the `N=` at each label indicates the number of knockdown-sensitive exons. The full dataset size is twice that number.

5.2 Is SpliceAI sensitive to RNA-binding proteins motifs?

In this section, we use the paired datasets generated to study SpliceAI in a per-RBP manner.

5.2.1 MutSplice: targeted *in silico* sequence perturbations to interpret SpliceAI

We aimed to investigate how the presence of known RBP binding motifs influences SpliceAI predictions. In other words, we wanted to perform a model sensitivity analysis to measure the

impact of perturbing sequence features on SpliceAI outcome. To do so in a scalable manner, we developed a pipeline called MutSplice (Figure 5.6). MutSplice is a modular pipeline that allows performing *in silico* sequence perturbations on a given set of sequences, running SpliceAI inferences over them, and processing the output to be ready for downstream analysis. The perturbations are not random; rather, they are guided by the location of known motifs in the sequences, hence being domain-oriented.

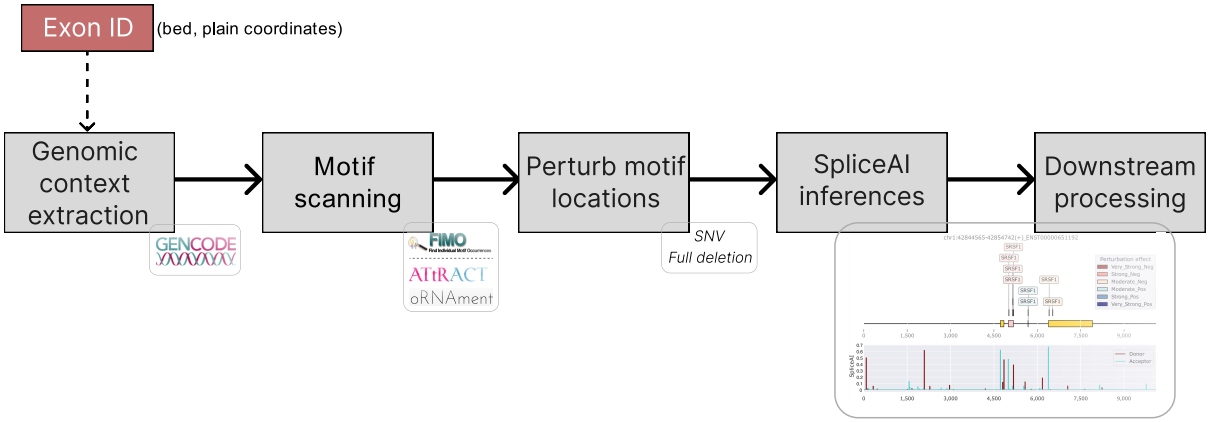

Figure 5.6: Schematic representation of MutSplice pipeline.

Specifically, the MutSplice pipeline works as follows:

- **Input** - MutSplice accepts exon coordinates either in bed format or standard plain genomic coordinates (chr:start-end). All additional preprocessing steps are handled internally.
- **Genomic context extraction** - MutSplice uses an internal transcript-aware genomic cache to assign a transcript ID for each exon, considering properties such as the MANE or CCDS status. It then extracts the genomic context, including surrounding introns and exons, while tracking splice site indexes. This process establishes an exon triplet, which serves as the sequence used for the analysis.
- **Motif scanning** - MutSplice scans the sequences for the presence of known RBP binding motifs using FIMO [215]. Several custom and public motif databases are available, such as ATrRACT [216] or oRNAment [217]. The output is a list of statistically significant motif locations within the sequences. These locations serve as potentially informative features of the model.
- **Perturb motif locations** - Perturbations in the sequences are performed at motif locations identified in the previous step. MutSplice supports two types of perturbations: full motif deletion, and an SNV at each motif position.
- **SpliceAI inferences** - MutSplice runs SpliceAI inferences over both the original and perturbed sequences. Besides recording predictions at splice site positions of the exon

triplet, MutSplice also saves high predictions at unannotated positions as they are putative cryptic splice sites.

- **Downstream processing** - Raw SpliceAI predictions are processed to quantify perturbation effects, further incorporating annotations like perturbation location, distances to splice sites, and motif density. Additionally, some visualizations are generated to help interpret the results.

MutSplice was implemented for the specific purpose of this chapter, and thus it is not a production-ready software. However, it is designed to be flexible and contains a set of parameters that allow analysis that extend beyond the scope of this chapter. MutSplice is implemented in Python and is available at `https://github.com/PedroBarbosa/mutsplice`.

5.2.2 Motifs exert a greater impact on SpliceAI predictions in knockdown-sensitive exons

For each paired dataset representative of a single RBP knockdown, we ran MutSplice to perform *in silico* perturbations at motif locations of the target RBP. For example, for the paired dataset of the SRSF1 knockdown, we remove the regions from the sequences matching the binding motif of SRSF1 to evaluate their marginal effects on SpliceAI predictions of the cassette exon. In simpler terms, these are ablation experiments where potentially informative features (constructed from the motif scanning procedure) are removed from the input sequence.

Like before, we proceeded with the knockdown experiments involving a minimum of 40 differentially spliced exons (N=49, Figure 5.2). In addition, some paired datasets referring to RBPs lacking binding motifs in the motif database were omitted from the analysis. This exclusion was due to the inability to ablate the motifs of the target RBP, as there were no PWMs available, rendering these datasets ineligible for this particular analysis. Among the available motif databases incorporated in MutSplice, we opted for ATrRACT [216], as it contained a greater variety of RBP motifs, resulting in 31 paired datasets for further analysis.

First, we examined the motif occurrences between each exon group, prior to the ablation studies. We expected knockdown sequences to harbor more functional motif occurrences than control sequences, given that their alternative splicing nature likely depends more heavily on such regulatory sequences. However, we observed equivalent motif proportions for most paired datasets between the two exon groups (Figure 5.7A). This observation is likely due to a high noise-to-signal ratio in motif occurrences, as many motifs are not biologically functional. Unsurprisingly, the number of occurrences (and their frequency relative to the dataset size) was highly heterogeneous across RBPs. Ubiquitous, highly expressed core splicing factors such as U2AF2, SRSF1, and hnRNP family members [218] showed the largest average number of motifs per sequence (Figure 5.7A, numbers on the y-axis).

When we analyzed the motifs whose ablation impacted SpliceAI predictions of the cassette

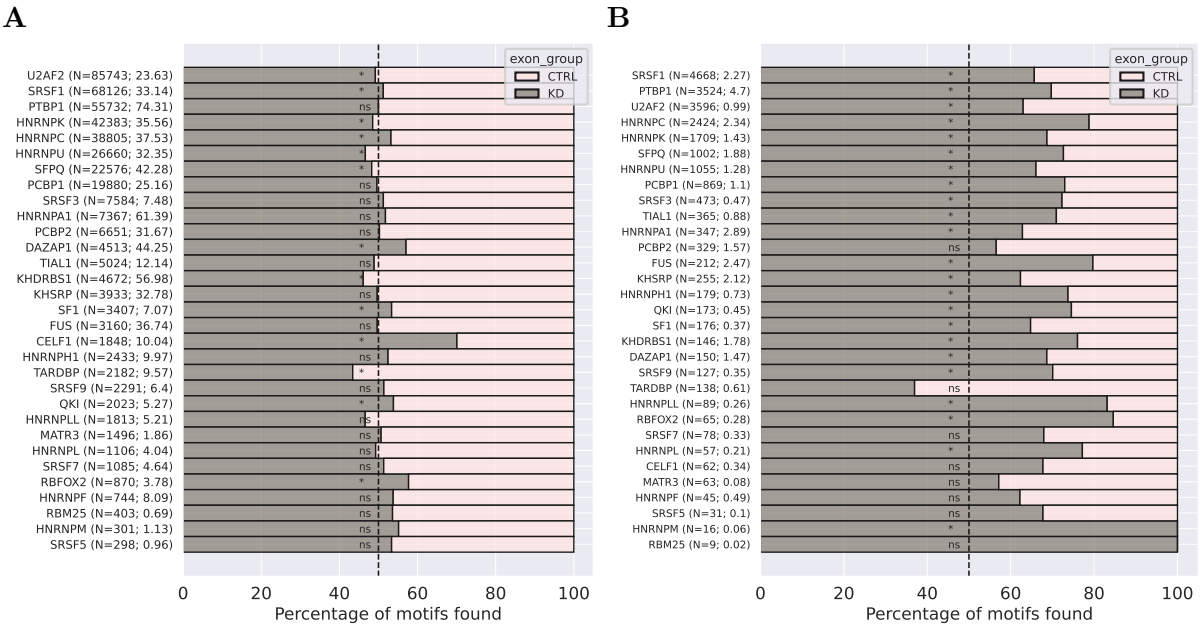

Figure 5.7: Percentage of motif occurrences from each exon group in each paired dataset. Integers next to the datasets labels refer to the total number of occurrences, while floats stand for the number of motifs per sequence (total number of motifs/paired dataset size). Statistical significance of motif occurrences between the two exon groups was assessed using a binomial test, which tested the null hypothesis that the probability of success (sampling a motif from a sequence of the knockdown group) is 0.5. p-values were corrected for multiple testing using Bonferroni method, at a family-wise error rate of 0.05. **A** - All motifs detected by FIMO (p-value < 0.0005). **B** - Motifs impacting SpliceAI predictions at the cassette exon positions. An impactful motif perturbation is defined as a model change $> |0.05|$ at the donor or acceptor positions compared with the original sequence.

exon (donor or acceptor change $>|0.05|$), we observed a different landscape. As we hypothesized, a larger proportion of these impactful motifs belonged to the knockdown group (Figure 5.7B). This finding indicates that despite the noise embedded in sequences, SpliceAI can detect signals consistent with expected biology. Specifically, alternatively spliced exons, which have suboptimal splice sites (Figure B.6), are more likely to be regulated by splicing factors [218]. In contrast, control exons, predicted with probabilities close to 1 (indicating constitutive splicing), showed resilience against RBP perturbations, with fewer ablations resulting in a model change compared to knockdown exons (Figure 5.7B, except for TARDBP).

Notably, the number of impactful motifs was severely reduced compared to total motif occurrences (Figure 5.7B, numbers on the y-axis). This confirms that only a small portion of the motifs are likely biologically functional, assuming the model accurately represents biology (Figure 5.8). These observations can be discussed from two perspectives: Technically, we already expected most perturbations at putative motif locations to be irrelevant since we deliberately set a fairly relaxed p-value threshold ($p < 0.0005$) to consider a motif hit as significant - a threshold

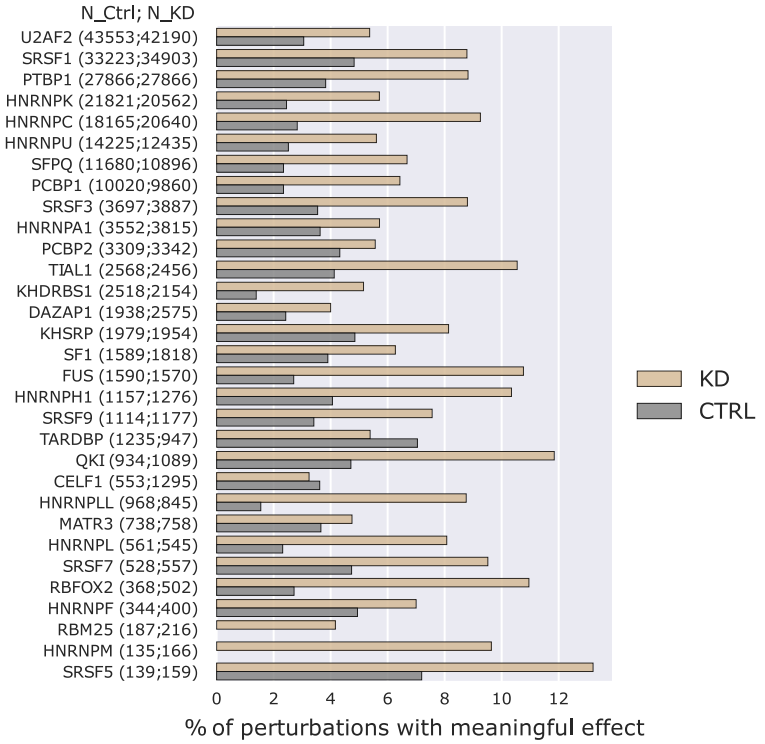

Figure 5.8: Percentage of motif perturbations with impact on SpliceAI predictions at the cassette exon positions, for each exon group. Numbers next to the dataset labels refer to the total number of motifs detected for each exon group. The percentages were calculated with respect to these numbers.

commonly accepted in sequence analysis. Consequently, we are inherently accepting a high false positive rate because our sequences are relatively long (over 10000bp to conform with SpliceAI model input specifications), leading to thousands of motifs being considered significant purely by chance [219]. Additionally, for short motifs (e.g., five nucleotides) with lower information content, even a perfect match may not be statistically significant if using a too stringent p-value threshold, hence eliminating potentially true biological motifs [215]. Our approach for model ablations prioritized sensitivity over specificity, as we were interested in capturing as many hypothetical true motifs as possible. Biologically, there are also multiple reasons for observing such a low percentage of impactful motifs. Introns, for example, are large and contain RNA binding motifs sparsely distributed in DNA. While some sequence elements may be identical to known motifs, their functional relevance is heavily influenced by factors such as cell type [202, 220], RNA structure [221–223], external stimuli [224, 225], and the surrounding sequence context [222].

Taken together, we show that SpliceAI predictions are sensitive to the presence of known RBP binding motifs, and this sensitivity is more pronounced in exons that were alternatively

spliced upon RBP knockdown. In addition, the small fraction of impactful motifs suggests that the model may have learned specific DNA syntax that distinguishes actual binding motifs from background noise.

5.2.3 Long-range sequence features influence SpliceAI predictions

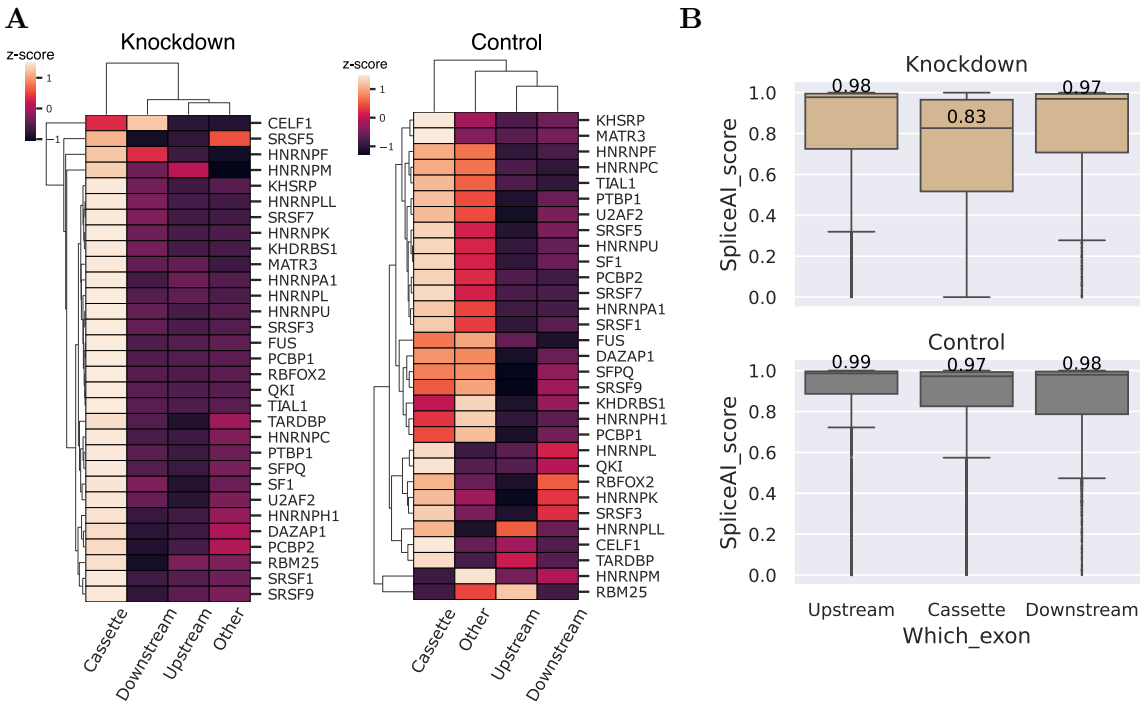

Figure 5.9: Perturbations mainly affect cassette exons, regardless of perturbed motif location. **A** - Heatmaps displaying the relative enrichment of impactful perturbations across different regions of the input sequences, for all paired datasets. The values represent the z-scores of the fraction of all perturbations impacting the splicing prediction of each considered region. Left heatmap: knockdown-sensitive sequences (24634 perturbations across all datasets and regions); Right heatmap: control sequences (N=15986 perturbations across all datasets and regions). Columns represent ablations affecting cassette, upstream, downstream exons, or any "Other" input position. **B** - Distribution of SpliceAI scores (average of predictions at the splice acceptor and donor positions) for exon triplets in all paired datasets. Top: Exons of knockdown-sensitive sequences. Bottom: Exons of Control sequences. The number on each box represent the mean value of the SpliceAI scores. Whiskers extend to 1.5 times the interquartile range.

Because SpliceAI uses a much larger sequence context than any previous model, we aimed to investigate whether the model indeed uses long-range features. To do this, we examined the prevalence of spatially distant, impactful perturbations on the splicing of cassette exons. Initially, we examined all perturbations affecting the prediction of any given position in the input sequence (prediction change $> |0.05|$), irrespective of their proximity to the cassette exon. Interestingly, we observed an enrichment of model ablations affecting the cassette exon compared to other

input regions (Figure 5.9A), particularly pronounced in sequences of the knockdown group. This underscores that model ablations predominantly influence the splicing of cassette exons, irrespective of the motif's proximity to the flanking exons or existing cryptic splice sites. By examining the SpliceAI predictions of the upstream and downstream exons in original sequences (with no ablations), we found they are predicted with close-to-1 probability (Figure 5.9B), indicating their lower susceptibility to ablations. This finding supports the idea that alternatively spliced exons are flanked by exons with optimal splice sites.

Next, we examined how much single ablations impact each splice site of the cassette exon. The high correlation between acceptor and donor motifs becomes evident, as perturbations affect both splice sites similarly (Figure 5.10A). This observation implies that SpliceAI does not use the local splice acceptor or donor motifs alone; instead, it learned the rules of exon definition by considering an optimal distance between the splice sites. This property was already explored in the original SpliceAI publication [38]. We further demonstrate the long-range feature dependency by plotting the number of impactful perturbations as a function of the distance to the cassette exon. While most impactful perturbations are located at <200bp from the splice site, we detected impactful perturbations as far as 3000bp away from the target exon (Figure 5.10B).

Analysis from a region-based perspective (rather than distance-based) revealed further insights (Figure 5.11). As expected, the strongest perturbation effects occurred when the ablated motifs overlapped with the exon borders and destroyed (or weakened severely) their splice sites. Moreover, effect sizes tended to be negative, meaning that motif ablations often decrease splicing probability (same trend visible at Figure 5.10A). This also aligns with our expectations, given that cassette exons in our paired datasets are typically predicted with probabilities closer to 1 than 0 (Figure 5.9,Figure B.6), implying that in many sequences perturbation effects can only be negative. Another interesting observation is that SpliceAI is sensitive to ablations that locate in the exons flanking the cassette exon and their upstream/downstream regions (so-called Intron_upstream_2 and Intron_downstream_2 regions), suggesting that SpliceAI, through its long input context, implicitly uses exon determinants of flanking exons to influence cassette exon splicing probability [226].

5.2.4 SpliceAI picks known binding rules of hnRNP and SR splicing factors

While previous analysis focused on an overview of SpliceAI sensitivity to distant features using all paired datasets, we now delve into context-specific effects of individual RBPs for which the binding rules were studied before. We first study the effect of RBP motifs on the SpliceAI score, based on whether the binding motif is located within exons or introns.

We investigated six hnRNP proteins with the greater number of impactful perturbations and observed that splicing-changing motifs mostly occur at introns (Figure 5.12). This is consistent with the knowledge that hnRNP proteins primarily bind to intronic regions to regulate splicing

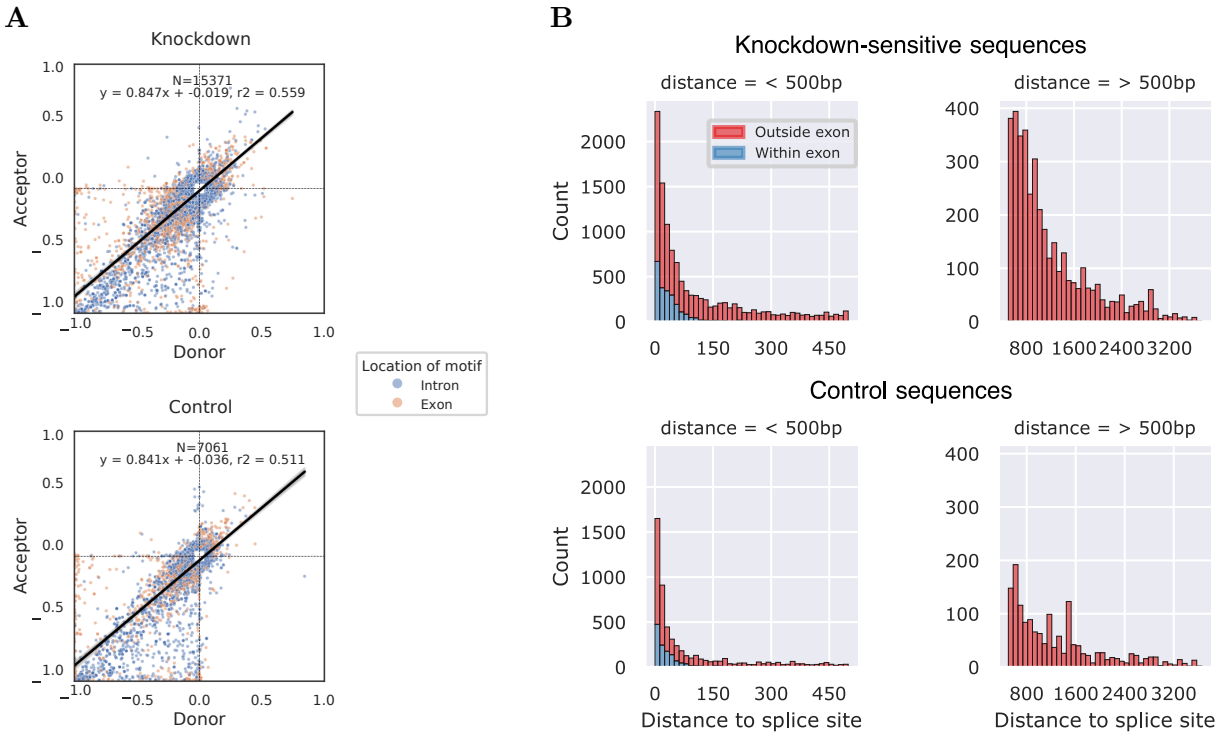

Figure 5.10: Distance-based perturbation effects on the splicing of the cassette exon. **A** - Perturbation effects on the splice donor and acceptor. "N" denotes the number of impactful perturbations, each representing one data point. The parameters of a fitted regression line are displayed, along with the coefficient of determination R^2. The color of the points indicates whether the ablated motif locates in intronic or exonic regions. Top panel: Knockdown-sensitive sequences. Bottom panel: Control sequences. **B** - Distribution of impactful perturbations as a function of the distance to the cassette exon ($min(distance\_acceptor, distance\_donor)$). Top panel: Knockdown-sensitive sequences. Bottom panel: Control sequences.

[43, 227]. We observed antagonist effects with hnRNP intronic motifs promoting both the increase and decrease of splicing predictions by the model, supporting the complex positional and context dependency of intronic motifs on exon activation and repression [47]. Although at a much lower frequency, we observed impactful binding sites at exons. Intriguingly, for HNRNPH1 this frequency was unusually high, with a trend of HNRNPH1 motifs promoting stronger exons, contrary to the classical role of hnRNP proteins repressing exon inclusion when bound to exons [43, 113].

As for the SR proteins, we detected a high prevalence of impactful motifs within exons that strengthen the probability of cassette exon splicing (Figure 5.13). This aligns with the known role of this class of proteins in promoting exon inclusion when bound to exons [42, 43]. Intronic impactful motifs revealed a more complex pattern. While the numbers are similar to exons, the frequency is proportionally much smaller given the larger size of introns. We similarly observed

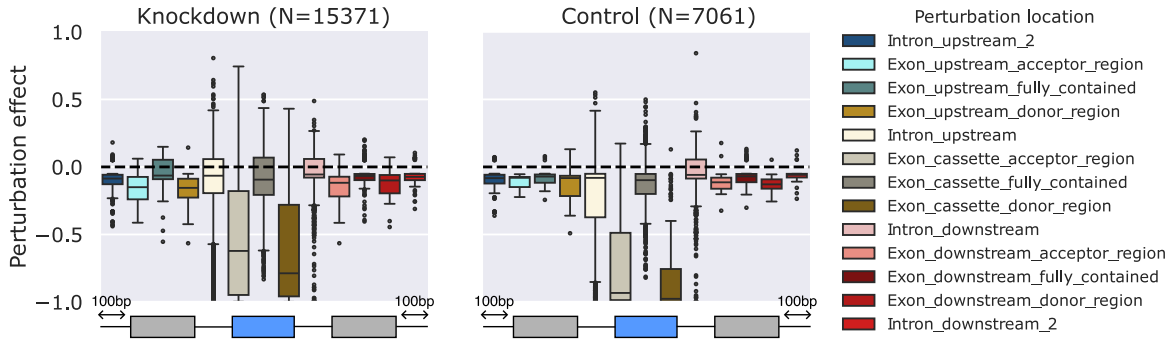

Figure 5.11: Region-based perturbation effects on the splicing of the cassette exon for knockdown-sensitive (left) and Control (right) sequences. Boxplot whiskers extend to 1.5 times the interquartile range. Exon's Acceptor region and Donor regions refer to ablated motifs that either overlap or are in proximity (< 2bp) to the splice acceptor and splice donor, respectively.

positive and negative splicing effects, corroborating previous studies highlighting contradictory functional consequences of SR intronic binding [228, 229].

To further disentangle the positional preferences of individual RBPs, we produced RBP maps [230]. Traditionally, these maps are generated by looking at alternatively spliced exons (and its flanking sequences) identified with RNA-Seq data (similar to what we did in Section 5.1.1 to extract knockdown-sensitive sequences) and then use the binding profile of the target RBP obtained from other data modality (e.g, peaks from eCLIP data) to infer positional rules by which the target RBP regulates alternative splicing. Here, to study the model we split the impactful perturbations by its effect (negative and positive), and used their location as a proxy for functional RBP binding.

When focusing solely on negative-effect perturbations to the model for the same subset of hnRNP and SR proteins, we observed enrichment profiles resembling the classical SR proteins: cassette exon enrichment of enhancer motifs (Figure 5.14A), as indicated by a decrease in SpliceAI score after their removal. Enrichment profiles of hnRNPs (except for HNRNPU) were scattered across upstream and downstream intronic regions, extending approximately up to 250bp into the intron. Notably, we observed stronger values at the first bin (-50bp) of the upstream intron for HNRNPA1, PTBP1, HNRNPC, and HNRNPK, and at the first bin (+50bp) of the downstream intron for HNRNPU (Figure 5.14A). For the former set of hnRNPs, and after additional inspection, the enrichment at -50bp was due to T-rich motifs at the polypyrimidine tract (-5bp to -20bp) that were probably important for the definition of the cassette exon (SpliceAI probability decreased after their removal). These results highlight a limitation of these analyses. Since several RBPs are known to bind the polypyrimidine tract, it is not possible to fully determine the contribution of the target RBP to that splicing event without incurring additional confounding effects. What we can infer is that SpliceAI is particularly sensitive to the

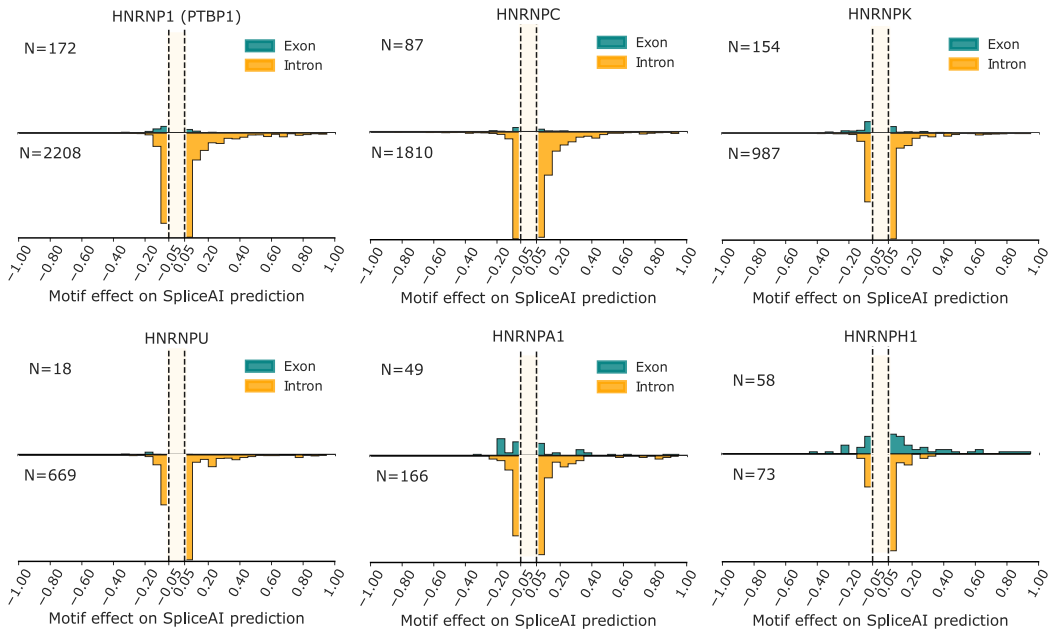

Figure 5.12: Context-specific effects of motifs from six hnRNP proteins on SpliceAI predictions in knockdown-sensitive sequences. Each 'N' represents the number of ablations influencing the model outcome (prediction difference $> |0.05|$). Ablation effect signs are inverted for clearer interpretation, reflecting the marginal effect of motif presence on the model outcome.

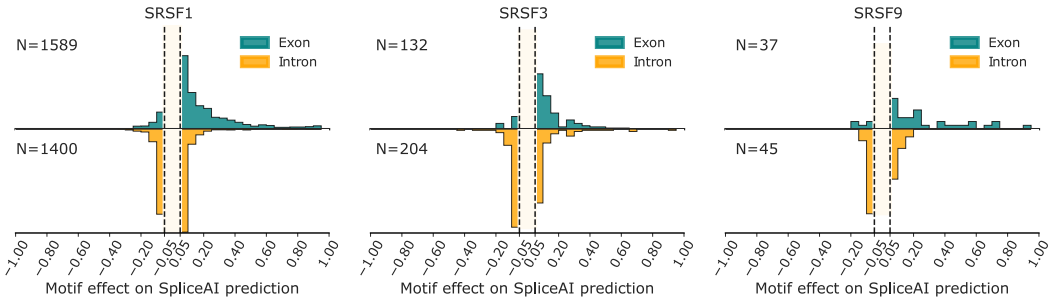

Figure 5.13: Context-specific effects of motifs from three SR proteins on SpliceAI predictions in knockdown-sensitive sequences. Each 'N' represents the number of ablations influencing the model outcome (prediction difference $> |0.05|$). Ablation effect signs are inverted for clearer interpretation, reflecting the marginal effect of motif presence on the model outcome.

presence of such motifs upstream of cassette exons. Conversely, control exons appear resilient to their ablation, likely due to strong splice site consensus motifs. That is what enrichment means in this context. On the other hand, HNRNPU enrichment at the downstream intron (Figure 5.14A) seems to be more specific, as most motifs were located between +30 and +50bp, potentially serving as intronic splicing enhancers (SpliceAI probability decreased upon their

removal).

For positive-effect perturbations, we obtained exonic enrichment of HNRNPA1, HNRNPU, or HNRNPK hnRNP proteins, supporting their classical roles as splicing repressors when bound to exons (SpliceAI score increased upon their removal, see Figure 5.14B). Regarding HNRNPC and PTBP1, profiles resembling those of negative-effect perturbations were observed (enrichment at upstream intron), implying that not only the position of the motif is important but also its surrounding context, where epistatic interactions with other motifs may occur to drive opposite effects on model predictions [43]. SR proteins showed distinct enrichment profiles, as they no longer cluster together (Figure 5.14B). This is consistent with SR proteins' complex, non-linear regulation landscape when not promoting exon inclusion [228].

RBP enrichment profiles were generated for all RBPs (Figure 5.14 C, D). While interpretation

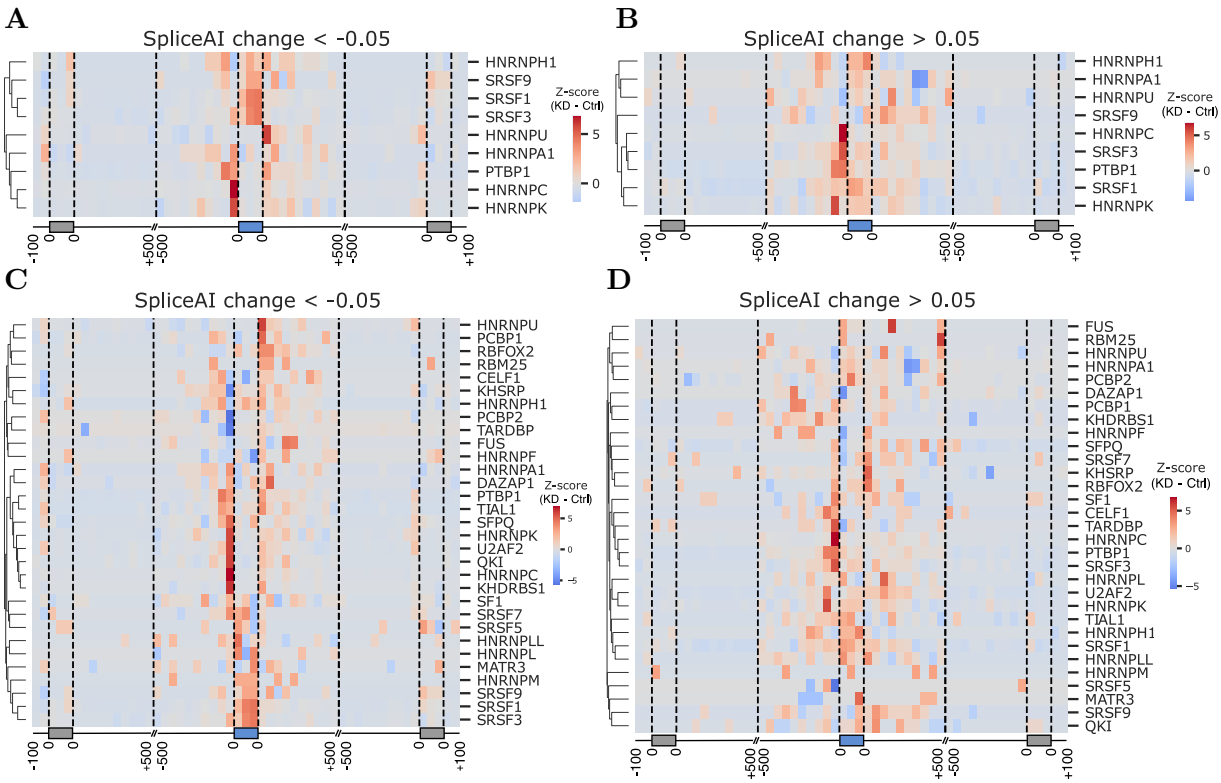

Figure 5.14: Position-dependent enrichment of impactful perturbations in knockdown-sensitive versus control sequences for all paired datasets. The distance to the upstream/cassette/downstream exons was discretized into region bins of 50bp, and the difference in the counts of impactful perturbations between sequences of the knockdown-sensitive and control groups was calculated for each bin. Heatmap values indicate the extent to which the differences in a specific region deviate from the mean of differences across all regions, for each RBP (row-wise). **A-B** - Perturbations with negative and positive effects on model predictions for a subset of hnRNP and SR RBPs. **C-D** - Perturbations with negative and positive effects on model predictions for all RBPs.

rules for all RBPs are beyond the scope of this chapter, we note, for example, a strong enrichment of FUS motifs deep in the downstream intron with opposing splicing effects, as previously observed in [231]. We also introduced another axis to the analysis by splitting it based on the direction of the effect observed in actual cells (sequences with more exon skipping - Figure B.7A - or inclusion - Figure B.7B - upon RBP knockdown). The complexity of the results increases even further. However, it was interesting to observe SpliceAI capturing the previously reported correlation of RBFOX2 and QKI binding profiles at the intron downstream of skipped exons upon RBP knockdown [113] (Figure B.7A, RBFOX2(-), QKI(-) rows, reflecting lower SpliceAI scores upon motif ablation).

Taken together, we demonstrate that SpliceAI partially reflects known binding rules of hnRNP and SR splicing factors. However, we reveal other complex and unexpected patterns (e.g., HNRNPH1) that warrant further investigation.

5.2.5 Interpretable tabular machine learning fails to emulate SpliceAI

We previously demonstrated the high predictive performance of SpliceAI in assessing the impact of genetic variation (Chapter 4). In this chapter, we studied alternative splicing and revealed SpliceAI's ability to differentiate between knockdown-sensitive and control sequences. Moreover, we showed that SpliceAI leverages RBP binding motifs as informative features for predictions. Given SpliceAI's discriminative power and the previous relative success of tabular machine learning models on predicting splicing levels using hexamer counts - k-mers of length 6 - as features [119, 232], we sought to assess whether such simpler, and interpretable models could be trained to mimic SpliceAI and contribute to its understanding.

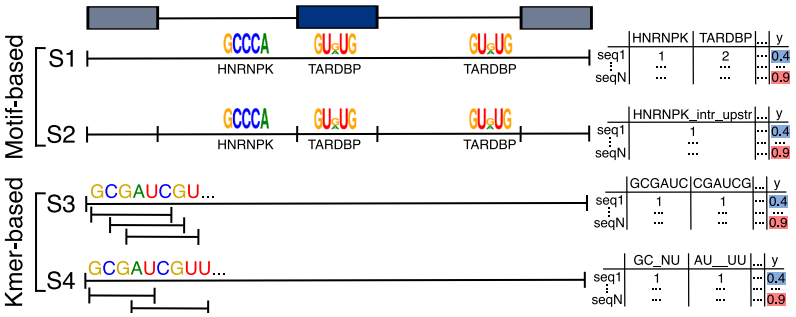

Figure 5.15: Engineering sequence-based features for tabular machine learning. Four different strategies to generate sequence-based features for predicting SpliceAI score. S1 and S2 strategies use as features motif occurrences obtained by running FIMO over the sequences against the ATrRACT motif database (hits with a p-value < 0.0001 were considered). S3 strategy uses fixed-length hexamer counts, while S4 strategy uses gapped k-mer counts obtained by mapping the sequences to a feature space using a kernel function. All counts were normalized to account for differences in sequence length.

We engineered four different sequence feature representations (Figure 5.15) to train regression

models to predict SpliceAI scores, for each paired dataset. The two first representations were derived from motif occurrences in the sequences at two levels of granularity (strategies S1 and S2 in Figure 5.15). The first was based on motif counts across the entire sequence, while the second across different regions of the exon triplet (as defined in Figure 5.11). Importantly, while in the ablation studies we were targeting motifs of the knockdown RBP, we now used motif occurrences of all RBPs as putative informative features, as binding motifs of other RBPs may contribute to the splicing outcome (e.g, additively or cooperatively). The last two representations are unbiased to known motifs as they come from k-mer occurrences (strategies S3 and S4 in Figure 5.15). We used both fixed length hexamer counts and variable length gapped k-mer counts.

The gapped k-mer representation is biologically motivated as it allows to capture the degeneracy of RBP binding by treating similar subsequences with mismatches and gaps as a single feature. For this special case, we employed a k-mer string kernel [233] to train Support Vector Machines (SVMs) on the input sequences. We used the LS-GKM package [234] tailored for regression problems [235]. The remaining feature representations (S1,S2,S3) were employed to train standard scikit-learn models, such as Linear Models and Random Forests.

Results showed that no feature representation and model combination could predict SpliceAI effectively (Figure 5.16), and that some configurations were arbitrarily worse than just predicting the mean of the dataset (the Dummy regressors). The best results were achieved by using gapped k-mers, although with low R^2 values. The max R^2 observed was 0.14 for the HNRNPK paired dataset. We also tried to frame the problem as a classification task, where rather than predicting the SpliceAI score, we predicted the exon group where a sequence belongs (knockdown-sensitive or control) using the same feature representations. Once more, results were poor, with several configurations performing as good as random guessing (Figure B.8).

It is worth noting that these tasks are challenging for such baseline models. The paired datasets share similar sequence composition between exon groups (that was the goal of generating these datasets, Figure 5.4), yet SpliceAI can predict alternative and constitute exons differently (Figure 5.9B). These findings underscore that simpler models are not able to approximate the DNN function. Even with the inclusion of gapped k-mers, fixed tabular features prove insufficient to capture the complex sequence patterns learned by the DNN. SpliceAI leverages spatial information embedded within sequences (even long-range features, as shown in Figure 5.10B) and such contextual information captures genetic interactions that tabular models simply cannot replicate.

5.3 Discussion and limitations

In this chapter, we conducted an interpretability analysis of SpliceAI through ablation studies involving known RNA binding motifs. We utilized specialized datasets tailored for this purpose, striving to mitigate the influence of other confounding factors on the model's predictions

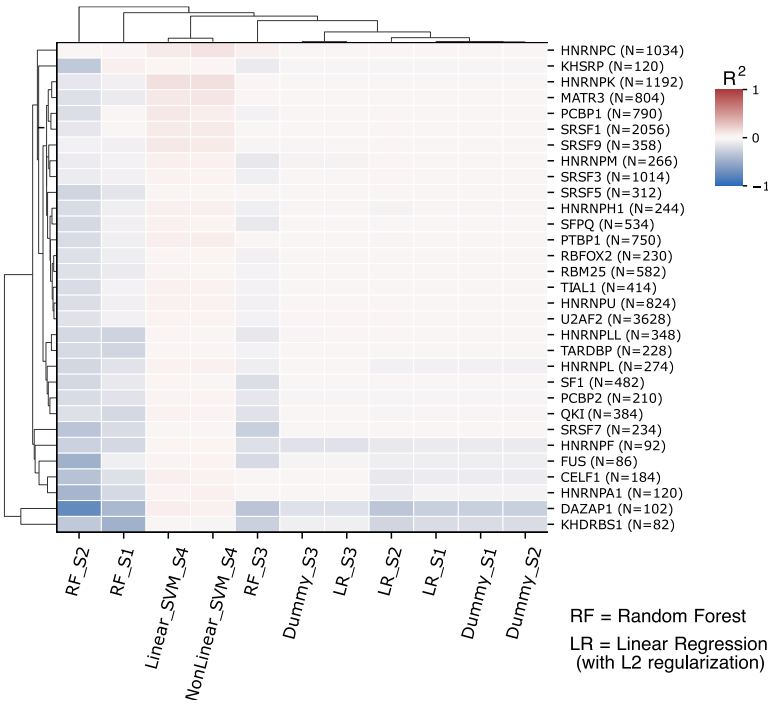

Figure 5.16: Predicting SpliceAI score for each paired dataset (rows) using several models and feature representations (columns). Performance was assessed by computing the mean coefficient of determination R^2 of test sets from a cross-validation procedure with 5 splits (and `shuffle=True, random_state=42`). Several models were used: a Linear Regression model with L2 regularization (Ridge regression with `alpha=1`), a Random Forest Regressor with `n_estimators=10` and two gkm-SVM models, one with a linear kernel (`gkmtrain -t2`), and another employing the RDF nonlinear kernel (`gkmtrain -t3`). Analysis includes also a baseline Dummy regressor that always predicts the mean of the dataset. The `N=` at each row label indicates the dataset size which includes the knockdown-sensitive and control sequences.

(Section 5.1.2). Through a comprehensive set of experiments, we found that SpliceAI predicts alternatively spliced exons with lower probabilities (Section 5.1.3) and that those exons are more sensitive to RBP binding motifs (Section 5.2.2). Furthermore, we have shown that SpliceAI indeed uses distant features to define an exon's probability (Section 5.2.3) and it appears to have learned, to some extent, positional rules of RBP binding and their effect on the splicing outcome (Section 5.2.4). Nevertheless, while these analyses provide valuable insights, several aspects deserve further discussion.

5.3.1 Experimental setup

The datasets used in this chapter were generated by processing RNA-Seq data from knockdown experiments performed in actual cells (Section 5.1.1). A gene knockdown experiment affects the overall expression of the target gene (in this case, an RBP), meaning that all the genomic

locations where the RBP binds to regulate splicing are, in theory, affected. However, in our computational experiments, we removed one binding motif at a time so that the marginal contribution of individual motifs could be assessed. Thus, this experimental setup does not reflect the actual biology of a gene knockdown. Joint effects of motifs could be evaluated by systematically removing all motifs simultaneously. However, depending on the number of motifs per input sequence, this could create unrealistic (and much shorter) sequences that would render model predictions unfaithful. Also, such a setup would make it harder to investigate positional dependencies of individual RBPs, as we have done with individual motif ablations.

5.3.2 Underestimation of effect sizes

It has been established that the contribution of *cis* regulatory sequences where RBP proteins bind is more subtle than that of the core splicing signals, such as the 3'SS, 5'SS and BP [43]. Indeed, most impactful RBP binding motifs had moderate-to-small effect sizes, with the distribution of ablation effects lying between |0.05| and |0.10| (Figures 5.12 and 5.13). Although this magnitude may correlate with the actual motif importance in cells, the effect size predicted by SpliceAI may be underestimated due to the neuron saturation problem of DNNs [30]. This is because multiple occurrences of the same motif may saturate the contribution of that feature, meaning that perturbing a single motif may not affect the output. In other words, multiple instances of the same motif in the input sequence may compensate for the ablation we performed. To counter this issue, we would need to run a reference-based attribution method like DeepLIFT [23], but that runs out of the scope of this chapter.

The other aspect to consider was already touched upon in Section 5.2.3. Since for most exons SpliceAI predicts splicing probabilities closer to 1 (mean of 0.83 and 0.97 for knockdown-sensitive and control exons, respectively, Figure 5.9B), the effect of impactful ablations tended to be negative. The effect of potential splicing-enhancing ablations (e.g., removal of splicing silencers from sequences) was most likely underappreciated since predictions cannot go above 1.

5.3.3 Motif analysis for putative feature construction

To identify potential features within sequences for perturbation, we scanned our datasets with motifs described as position weight matrices. In Section 5.2.2, we briefly addressed some challenges inherent of such classical motif analysis, notably the high false positive rate when scanning long sequences. Equally important is the underlying assumption of such analyses. We operated under the assumption that the motifs identified represent the actual biology, upon which all subsequent interpretation analyses were based. However, it's known that many RBPs have low sequence specificity [236] or their binding motifs are incompletely characterized [237]. Therefore, we cannot exclude the possibility that SpliceAI, trained blind to motif information, may have learned motifs that are yet to be discovered and were not queried in our experiments.

Moreover, there is the aspect of redundancy to consider. While some motifs are RBP-specific, others are shared across multiple RBPs [222]. This implies that caution is necessary when assigning feature importance to a single RBP. In our experiments, we partially controlled for this issue by conducting analyses on a per-RBP basis, where we ensured that the most of knockdown-sensitive exons were affected by only one or two RBP knockdowns (Figure 5.1B).

5.3.4 SpliceAI is cell-type agnostic

SpliceAI was trained on a large dataset of pre-mRNA sequences containing positions that are either a splice site or not. Hence, when a new sequence is fed into the model, SpliceAI predicts, for each position, the probability of it being a splice site (acceptor or donor). This classification setup captures the strength of a splice site (how confident the model is that a position is a splice site), but it is entirely agnostic to the usage of that splice site in a specific cell type. For example, we can have the strongest splice site in the world, but if the gene where it belongs is not expressed in the cell type under study, its usage is 0.

This is a limitation of the model, as the splicing outcome is highly dependent on the cellular environment (e.g., which RBPs are present and at which concentrations) [46]. In our experiments, we used exons that were affected by RBP knockdowns in the HepG2 cell line. When examining the relationship between the dPSI and perturbation effect predicted by SpliceAI, we observed no meaningful correlation (Figure 5.17). It shows that many perturbations predicted by SpliceAI had the opposite effect compared to what was observed in cells. This outcome was somewhat predictable, as the model was not specifically trained to predict dPSI. While SpliceAI excels as an end-to-end prediction model of RNA splicing, care should be taken when interpreting splicing changes in specific cellular contexts.

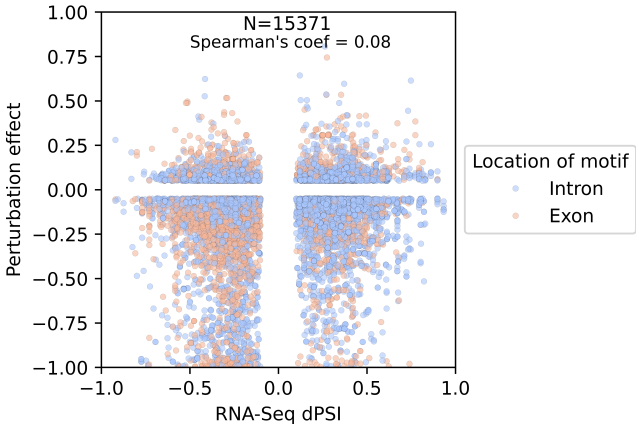

Figure 5.17: Relationship between SpliceAI perturbation effects (prediction change > |0.05|) in knockdown-sensitive sequences and the deltaPSI observed in RNA-Seq data. In the cases where multiple impactful perturbations were observed for a single exon, the same deltaPSI value was assigned to all of them.

5.3.5 Deciphering new biology

In this chapter, we perturbed sequence features and measured how SpliceAI predictions changed. We then performed *post-hoc* interrogations through various perspectives to understand the model's behavior. While this overview proved useful to pinpoint SpliceAI's ability to learn known biology, we now question whether such global analyses might limit its potential to uncover new biological insights.

SpliceAI learned a highly complex representation of the RNA splicing mechanism, and our attempts to construct simpler, interpretable surrogates to replicate its behavior were largely unsuccessful (Section 5.2.5). While one can argue that we did not tune the models properly (we did not perform hyperparameter optimization), we believe that no amount of tuning would make a big difference. Instead, we wonder whether approximating the DNN function within local regions of the sequence space might be beneficial for generating new biological hypothesis. While the ultimate goal is to unveil general regulatory mechanisms and rules, the reality is that each exon's splicing may have its own specifics, given the vast combinatorial space of sequence elements that can interact and influence the final splicing outcome [238]. Indeed, splicing research has historically been conducted on individual exons [211], a practice that continues today (e.g., [239]). Therefore, in the next chapter, we explore strategies to study deep learning models and splicing locally at the individual exon level.

Chapter 6

Semantically-rich synthetic dataset generation with constrained Genetic Programming

This chapter establishes the groundwork for employing evolutionary-inspired algorithms integrated with domain knowledge through grammars, with the goal of exploring the local behavior of sequence-based deep learning models. In particular, we discuss deep learning-guided strategies for sequence generation and propose a GP approach for the task of generating semantically rich synthetic datasets for explainable AI. Contributions of this chapter include:

- Synthetic data generation with a GGGP approach where grammars restrict the search space by encoding RNA splicing knowledge. We extend the GeneticEngine framework to support genomics applications, particularly by designing custom callbacks and meta-handlers. We demonstrate the effectiveness of this approach by generating synthetic sequences that span the whole prediction landscape of SpliceAI, with a significant improvement over random search-based strategies.

  Paper: **P. Barbosa**, R. Savisaar, A. Fonseca, "Semantically Rich Local Dataset Generation for Explainable AI in Genomics", *GECCO '24*, 2024 [40].

  Reproducibility: https://github.com/PedroBarbosa/Synthetic_datasets_generation

- Contributed to the evaluation and validation of Genetic Engine, a framework developed in our group that implements GGGP.

  Code: https://github.com/alcides/GeneticEngine

  Paper: G. Espada, L. Ingelse, P. Canelas, **P. Barbosa**, A. Fonseca, "Data types as a more ergonomic frontend for Grammar-Guided Genetic Programming", *GPCE '22: Concepts and Experiences*, 2022 [39].

  Reproducibility: https://github.com/pcanelas/GeneticEngineEvaluation

6.1 Introduction

In the previous chapter, we explored sequence perturbations as a strategy to interpret SpliceAI. We demonstrated that some position-dependent patterns for known motif features could be recapitulated globally (Section 5.2.4). However, taking a global perspective of the model's sensitivity to such features, coupled with the complexity of the RNA splicing domain, may have hindered our ability to identify local sequence patterns important for model prediction. In addition, we had limited data for a couple of RBPs to conduct a robust global analysis (Figure 5.2), making a local analysis of the sequence space an appealing alternative.

Attribution methods (Section 3.4.2) could be employed to measure position-specific nucleotide effects and, therefore, provide local explanations. However, these methods hold some limitations. Results can differ according to the attribution method used [240], and epistatic feature interactions are not quantified [30]. An alternative strategy is to generate local synthetic datasets around a single input. These datasets harbor sequences with combinations of perturbations, which can be used to train interpretable surrogate models on such localized regions of sequence space. While the concept of local surrogates has been proposed for other domains [31–33], its application to genomics has been proposed only recently [34], and not for the RNA splicing problem.

For such surrogates to succeed, it is important that the local datasets contain sequences that are not only syntactically similar but also semantically diverse, so as to extensively sample the model fitness landscape. The generation of such a local dataset is a challenging task, considering the vast combinatorial search space and the irregular fitness landscape. It requires exploring the syntactic neighborhood of the target sequence and identifying perturbed sequences that maximize the diversity of the semantic space. The search space grows linearly with the length of the sequence and exponentially with the number of simultaneous single nucleotide perturbations. It grows even faster when considering more complex, but realistic perturbations, such as insertions of deletions of short lengths. As a result, techniques such as exhaustive search or random search may not be feasible in this context. Exhaustive search becomes computationally intractable, while random search is oblivious to the semantic space, which may lead to datasets that sparsely cover the fitness landscape. We reason that some heuristics are needed to efficiently explore the search space, and we therefore frame this problem as a dataset generation task.

6.2 Deep learning-guided sequence generation

We now provide an overview of the state-of-the art methodologies that explicitly focus on synthetic data generation in genomics. In particular, we explore various applications of sequence generation guided by performant deep learning models.

One such application is sequence design, which involves designing molecules with desired outcomes, such as controlling gene expression levels [241] or developing more efficient mRNA vaccines [242]. To do this, deep generative networks are used to model distributions of sequences with desired properties [243–247]. These frameworks employ activation maximization to maximize the property of interest by gradient ascent through the model oracle, typically using an appropriate generative network based on Generative adversarial networks (GANs) [248] or Variational Autoencoders (VAEs) [249]. Genetic algorithms have also been proposed, either by greedily perturbing the best sequences [250], employing very large population sizes [251] or evolving ensembles of optimization algorithms [251]. Recently, inspired by classical wet lab experiments, directed sequence evolution has been used successfully to iteratively evolve a random sequence into a synthetic, biologically functional sequence [252, 253]. This method exhaustively measures the impact of SNVs at each iteration and selects the perturbation that yields the largest model prediction change to be applied to the sequence. However, it's important to note that despite their utility, these approaches have inherent constraints. They do not generate sequences covering the entire fitness landscape of the oracle (the semantic space). In addition, many methods were not developed for model explainability, and can only scale to short sequence contexts.

The use of synthetic data augmentations has been proposed to enhance model generalization and interpretability [34, 240, 254]. In Prakash *et al.* [240], sequence generation is guided by a motif-based pipeline to fine-tune trained models and benchmark different *post-hoc* explainability methods. In contrast, EvoAug [254] applies random augmentations to the sequences during pretraining, using the same label as the wild-type sequence. Then, the potential biases and labeling errors created with the augmentations are addressed by fine-tuning on the original data only. The adoption of these augmentations has demonstrated improved generalization and more interpretable feature attribution maps. Finally, SQUID [34] employs synthetic data generation to train interpretable surrogates to elucidate deep learning models in local regions of the sequence space. This approach aligns with our goal, where oracle predictions serve as labels for the synthetic dataset. However, in SQUID, perturbations are applied randomly without ensuring comprehensive coverage of the model's semantic space. This might be a limitation, depending on the application. For instance, in our RNA splicing case study, if one wants to study what drives the splicing levels of a wild-type exon from 0% inclusion to 100%, it is unlikely that the randomly generated synthetic dataset will adequately cover the two far-reaching locations of the semantic space.

6.3 Proposed approach

We propose using GP [103] to build local datasets generated from a single sequence. The evolutionary loop evolves sequences containing perturbations that trigger diversity in the

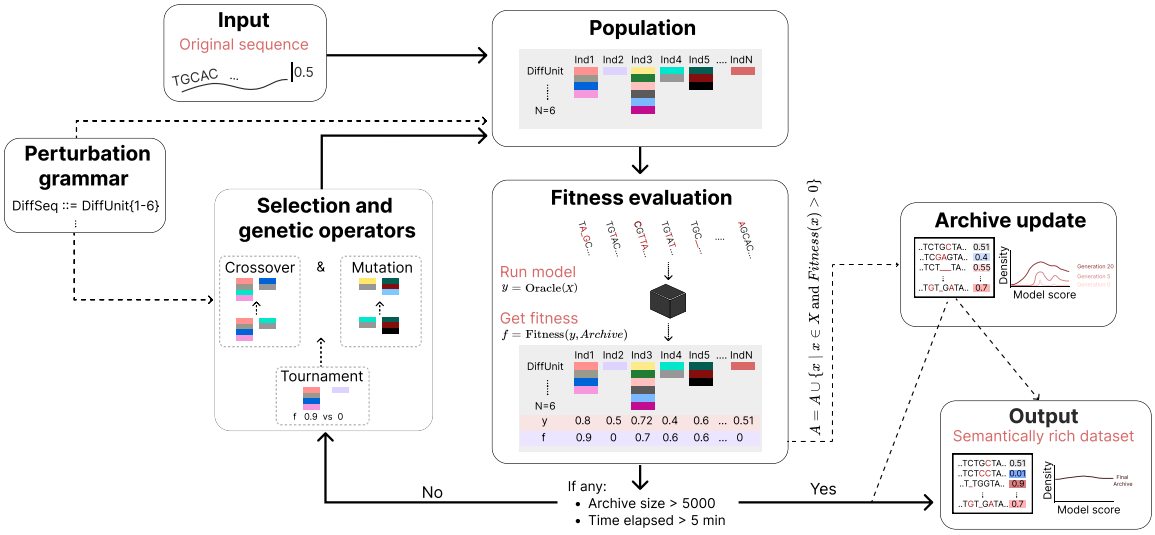

Figure 6.1: Summary of the proposed methodology.

semantic space of the deep learning model. Importantly, custom domain aware grammars restrict the perturbations applied to the original sequence, hence significantly reducing the search space.

GP explores the search space by maintaining a population of interesting programs or instances — in our case, combinations of perturbations — and by applying genetic operators such as mutation and crossover. Throughout the evolution cycle, relevant individuals are copied from the population to an archive. Once the evolution ends, either due to a time limit or reaching the archive capacity, this archive becomes the final dataset. Thus, the population is used to keep individuals that represent promising areas of the search space, not the final dataset (Figure 6.1).

We use a domain-specific representation for individuals. Rather than sequences as strings of nucleotides, we choose to represent the perturbations themselves using a grammar (Section 6.3.1). This representation draws inspiration from diff files, which compactly represent two similar files by listing only their differences. It offers several advantages, including reduced memory usage compared to storing two complete copies and being easier for practitioners to interpret. Additionally, genetic operators can work on the semantic level of a perturbation, allowing the introduction of domain-specific and biological constraints.

Consequently, the initial population is randomly generated using the semantic rules encoded in the grammar. This population undergoes evaluation by applying the perturbation representation to the original sequence, resulting in perturbed sequences. These sequences are then fed into the deep learning oracle to obtain predictions in the semantic space of the model.

Perturbation Sequence	`DiffSeq`	::=	`DiffUnit{1-6}`
Perturbation	`DiffUnit`	::=	`SNV`
		|	`Insertion`
		|	`Deletion`
SNV	`SNV`	::=	`SNV(Pos, Nuc)`
Insertion	`Insertion`	::=	`Ins(Pos, Nucs)`
Deletion	`Deletion`	::=	`Del(Pos, Size)`
Multiple Nucleotides	`Nucs`	::=	`Nuc+`
Nucleotide	`Nuc`	::=	`A`
		|	`C`
		|	`G`
		|	`T`
Position	`Pos`	::=	`int`
Size of deletion	`Size`	::=	`int`

Figure 6.2: The core structure of the grammar used to represent an individual in respect to the original sequence, presented in EBNF. Underlined symbols are terminals or meta-handlers.

Our goal is not to optimize the prediction itself, but rather the diversity within the archive's predictions (Section 6.3.2). To achieve this, we define two fitness functions (Section 6.3.3) that are computationally lightweight and can guide the population towards archive diversity. Finally, we follow a traditional GP loop, with tournament selection (size 5) and application of grammar-guided genetic operators such as tree-based crossover and mutation (Section 6.3.4). Additionally, we propose a custom mutation operator that promotes locality in the sampled positions of perturbations, acting as a more local search than the traditional tree-based mutation. As a case study, we focus on RNA splicing, particularly on exploring the prediction landscape of SpliceAI (Section 6.4).

6.3.1 Representation

We devised a perturbation grammar to constrain the perturbations to be biologically plausible. Using GGGP [104], extended with meta-handlers [39], both the population initialization and genetic operators modify the representation of individuals within these constraints.

Our grammar (Figure 6.2) defines individual genotypes as a sequence of 1 to 6 perturbations (`DiffSeq`, the starting symbol). Each perturbation (`DiffUnit`) can be one of three types: a `SNV(Pos,Nuc)` parameterized with a position and a new nucleotide, an insertion of a short sequence at a given position (`Insertion(Pos,Nucs)`), or a deletion of n nucleotides at a given position (`Deletion(Pos, Size)`). Table 6.1 provides examples of different perturbations and their effect on the resulting sequence.

Four principles guided the design of grammar: Firstly, the number of perturbations was

Table 6.1: Examples of perturbations and their effect on the original sequence.

Original Sequence	Perturbation	Final Sequence
ATTCGCGTTA	[SNV(1,A)]	AATCGCGTTA
	[Ins(2,CG)]*	ATCGTCGCGTTA
	[Deletion(7,2)]	ATTCGCGA
	[SNV(1,A), Insertion(2,CG)]	AACGTCGCGTTA

capped at 6 to preserve similarity to the original sequence. This restriction is important, as increased dissimilarity would decrease explanatory value. Secondly, the three types of `DiffUnit`s, despite their varying lengths, represent types of genetic variation found in nature. We used meta-handlers to restrict the length of deletions and insertions to a maximum of 5. Thirdly, we prevent overlapping perturbations as it is impractical to keep track of the correspondence between positions and the original sequence. In these cases, only the largest perturbation is kept, hence prioritizing insertions and deletions over SNVs. Finally, we impose additional constraints on exploring specific regions within the sequence based on problem-specific requirements. This is particularly relevant when exploring certain regions that could significantly impact the search process, potentially leading to local optima. For instance, in the context of the RNA splicing problem, we explicitly restrict any `DiffUnit` within the positions `[-10, 2]` and `[-3, 6]` around splicing acceptors (the start of an exon) and donors (the end of an exon), respectively. These restrictions are enforced using meta-handlers on the values of the `Pos` non-terminal.

Ultimately, the genotype is a list of non-overlapping perturbations applied to the original sequence.

6.3.2 Archive

The generation of the archive is the main outcome of the evolutionary algorithm. It is used as a dataset for any downstream application. The archive has a fixed specified capacity S and is composed of N equally-sized bins (or buckets). In this study, we used 5000 and 40 bins, each representing a range of 0.025 within the black box prediction space $\{p \in \mathbb{R} \mid 0 \leq p \leq 1\}$. The optimal archive would maintain an equal and maximum capacity in all bins while also displaying good diversity within each bin.

The quality (Q) of an archive A is the weighted sum of the number of sequences stored in the archive (A_{size}), the archive inter-bin diversity ($\hat{D}$), the intra-bin diversity ($\hat{D}_{per\_bin}$) and the fraction of bins with at least 10 sequences ($A_{no\_low\_count\_bins}$):

$$\begin{aligned} Q(A) = & 0.3 \times A_{size} \\ & + 0.3 \times \hat{D} \\ & + 0.2 \times \hat{D}_{Per\_bin} \\ & + 0.2 \times (A_{no\_low\_count\_bins}) \end{aligned} \tag{6.1}$$

The raw archive diversity D is quantified using the Shannon diversity index and normalized ($\hat{D}$) to scale between 0 and 1:

$$\begin{aligned} D(A,N) &= -\sum_{b=1}^{N} p_{A_b} \ln(p_{A_b}) \\ \hat{D}(A,N) &= \frac{D(A,N)}{ln(N)} \end{aligned} \tag{6.2}$$

where p_{A_b} stands for the proportion of the sequences in the archive belonging to the b^{th} prediction bin.

The intra-bin diversity D_{per_bin} measures the average diversity within each of the N bins by further dividing each archive bin (A_b) into 10 equally-sized sub-bins and calculating the diversity $\hat{D}(A_b, 10)$ for each. The final $\hat{D}_{per\_bin}$ is the average diversity across all bins:

$$\hat{D}_{per\_bin} = \frac{1}{N} \sum_{b=1}^{N} \hat{D}(A_b, 10) \tag{6.3}$$

Finally, the $A_{no\_low\_count\_bins}$ quantifies the fraction of bins with more than 10 sequences. These factors balance a semantic representation that is both coarse and fine-grained, ensuring an even distribution across all bins, and consider the total number of sequences in the final dataset. While other metrics could be considered, we chose these for their relevance to our case study.

6.3.3 Fitness Functions

The purpose of the fitness function is to assess how likely an individual is to be kept in the next generation. For example, even the worst individual of a given generation can be added to the archive if it helps improve its quality (fitness > 0). However, it will probably not survive for the next generation and its genotype will be lost.

We define two fitness functions that take into account the current archive status: *Bin Filler* and *Increase Archive Diversity (IAD)*.

Bin Filler: This fitness function is directly proportional to the number of available slots in the bin that the current individual i belongs to. It is defined as one minus the ratio between

the number of archive sequences in the bin b (p_{A_b}) and the target number of sequences per bin T. T is computed *a priori*, based on the desired archive size S and the number of bins N:

$$\begin{aligned} BF_i &= 1 - \frac{p_{A_b}}{T} \\ T &= \frac{S}{N}. \end{aligned} \tag{6.4}$$

It aims to promote the survival of individuals that belong to emptier bins. This enhances the exploration of a combination of perturbations in the least explored areas of the oracle semantic space.

IAD: This fitness function measures how much the addition of individual i to the Archive increases its inter-bin diversity, as described in Equation (6.2):

$$IAD_i = \hat{D}(A \cup i, N) - \hat{D}(A, N). \tag{6.5}$$

It is designed to be less reliant on the current filling of each bin. Instead, it assigns higher fitness to a sequence if it positively contributes to the overall archive uniformity at that moment. This strategy might be advantageous in avoiding being stuck on local optima since evolution favors sequences that fully deviate from them. Nevertheless, both fitness functions share the same overall goal.

6.3.4 Genetic Operators

We use standard tree-based GGGP mutation and crossovers, extended with meta-handlers [39]. In a typical GGGP mutation, a mutation at a given position of the list would generate random, new elements for the remainder of the list. Through the usage of meta-handlers, mutations on lists result in either adding, removing or replacing exactly one element.

We also designed a custom mutation operator that replaces a randomly chosen `DiffUnit` with another one in proximity. This replacement is determined by a normal distribution centered at the position of the old `DiffUnit`. This approach enhances the search for functional local motifs in the sequence, a known property of biological sequences. As an example, a typical mutation in a GGGP individual, like `[SNV(7, G)]`, could replace the node position 7 with a randomly sampled integer such as 8345. Our custom mutation would select the node and replace it with `[SNV(`v`, G)]`, with $v \sim N(7, 4)$.

6.4 Evaluation methodology

To assess our GGGP method for local dataset generation, we employ it to synthesize local datasets for explaining SpliceAI, a neural network that models RNA splicing (Section 6.4.1).

We describe the specific experimental settings, including hardware and software details, in Section 6.4.2. Additionally, in Section 6.4.3, we detail the baseline approach against which our methodology is compared. Finally, Section 6.4.4 details our process for tuning hyperparameters.

6.4.1 Case Study

While several applications for sequence generation do exist, our evaluation focuses on the problem of RNA splicing. In particular, we aim at generating synthetic datasets for local explainability of the SpliceAI model [38]. SpliceAI has shown remarkable success in the prediction of pathogenic variants [155, 156, 206], and we have provided the same evidence for deep intronic regions of the human genome (Chapter 4). In addition, SpliceAI predicts constitutive and alternatively spliced exons differently (Chapter 5), suggesting that the model has indeed learned, at least partially, mechanistic rules of the splicing code.

In this study, the input is a DNA sequence representing an exon triplet along with the intervening introns. The goal is to generate sequences that influence the probability of inclusion of the middle exon (the so-called cassette exon). In biological terms, we target the generation of sequences to model exon skipping, the most prevalent alternative splicing event in the human genome [255]. Although SpliceAI does not directly model PSI, a well-established metric for quantifying exon inclusion levels, we use the average of SpliceAI predictions at the acceptor and donor positions of the cassette exon as a proxy for PSI values (as done in Chapter 5). This decision is justified by the observed correlation with PSI measurements from RNA-Seq data [38].

The SpliceAI input is bounded to 10,001 nucleotides, ensuring that 5k of flanking context on both sides is considered for prediction of each central position undergoing evaluation. Sequences (exon triplets) shorter than this resolution were padded whereas sequences longer than 10k nucleotides were trimmed to conform with the model input dimensions.

As a proof-of-concept, we used the exon 6 of the FAS gene, an exon extensively studied [238, 256–258] due to the fact that excluding this exon switches the protein's function from pro-apoptotic (programmed cell death) to anti-apoptotic. In addition, the PSI levels of this exon vary across tissues and displayed intermediary PSI levels of 60% in a minigene construct containing exons 5-7 and the corresponding introns [258]. Similarly, using the same genomic context, SpliceAI predicts a PSI value of 0.4921 for exon 6, aligning with the observed behavior in real cells.

6.4.2 Experimental settings

Our approach and the baseline were implemented on top of GeneticEngine v0.8.5 [39], which supports meta-handlers that allow encoding constraints on the perturbations.

All the experiments were conducted on a Ubuntu 22.04 server with an AMD Ryzen Threadripper 3960X 24-Core Processor with 96GB of usable RAM. The GPU used for model

inferences was an NVIDIA GeForce RTX 3090 with 24Gb of VRAM, running on CUDA v12.3 and Python 3.10.12. Reproducibility instructions are available at https://github.com/PedroBarbosa/Synthetic_datasets_generation. The datasets generated in this study are available on Zenodo at https://doi.org/10.5281/zenodo.10607868.

6.4.3 Baseline

Existing work that generates local synthetic sequences employs either random [34] or exhaustive (from a short 500-nucleotide sequence [253]) sampling. Since exhaustive search is impractical for large search spaces like ours, we adopt Random Search (RS) as the baseline. It is worth noting that the baseline also takes advantage of our semantically rich encoding and does not operate on sequences directly, thus enabling us to focus the evaluation on the impact of the GP loop.

Both approaches are compared with the same time budget and are implemented on the same framework, reducing the impact of external factors in our evaluation.

6.4.4 Hyperparameter Optimization

We used Optuna v3.4.0 [259] for hyperparameter optimization of the evolutionary algorithm. The objective was to identify the optimal configuration that maximized archive quality, as in Equation (6.1). We used Tree-structured Parzen Estimator Approach (TPE) [260] for parameter sampling. The optimization process was carried out until 500 trials were successfully completed. Each trial was set to finish when either of the following conditions was met: the archive accumulated 5000 sequences, or the allocated time budget of 5 minutes was reached. We also added soft constraints on the sum of certain hyperparameters, favoring their total to be between 0.5 and 1. These constraints were applied to three sets of parameters: the sum of *SNV grammar weight*, *Insertion grammar weight* and *Deletion grammar weight*; the sum of *Genetic operators weight*, *Elitism weight* and *Novelty weight*; and the sum of *Mutation probability* and *Crossover probability*. The search space for all hyperparameters is reported in Table 6.2. Individual optimizations were conducted for each of the fitness functions, resulting in four distinct optimization runs: GGGP_BinFiller, RandomSearch_BinFiller, GGGP_IAD and RandomSearch_IAD.

6.5 GP with Bin Filler as fitness function performs best

Firstly, we compared the top five Optuna trials of each strategy throughout an evaluation using five random seeds (Figure 6.3A). We observed a high agreement across seeds within each strategy, except for a single trial with the IAD fitness function. The best overall configuration resulted from combining Genetic Programming with Bin Filler, both in the final quality of the archive and the rate at which it increases throughout the evolution.

Table 6.2: List of hyperparameters tuned by Optuna along with the best values for each strategy

Hyperparameter	Search space	GGGP_BF	GGGP_IAD	RS_BF	RS_IAD
Max *DiffUnits*	Int{1,2,3,4,5,6}	5	4	6	5
Max insertion size	Int{1,2,3,4,5}	5	5	5	5
Max deletion size	Int{1,2,3,4,5}	3	1	5	4
SNV grammar weight	Float[0, 1] (Step 0.05)	0.05	0.15	0.1	0.25
Insertion grammar weight	Float[0, 1] (Step 0.05)	0.75	0.25	0.85	0.4
Deletion grammar weight	Float[0, 1] (Step 0.05)	0.3	0.1	0.15	0.35
Population size	Int[100, 1900] (Step 200)	500	700	1300	1900
Selection method *	Cat{Tournament, Lexicase}	Tournament	Tournament	-	-
Crossover probability *	Float[0.05, 5] (Step 0.05)	0.25	0.2	-	-
Mutation probability *	Float[0.2, 1] (Step 0.1)	0.7	0.7	-	-
Use custom mutation operator *	Bool{True, False}	True	True	-	-
Custom mutation operator weight *	Float[0, 1] (Step 0.1)	0.8	0.7	-	-
Genetic operators weight *	Float[0, 1] (Step 0.1)	0.8	0.8	0	0
Elitism weight *	Float[0, 1] (Step 0.1)	0	0.1	0	0
Novelty weight *	Float[0, 1] (Step 0.1)	0.1	0	1	1

* In Random Search, these hyperparameters were not optimized. We strictly set Novelty weight to 1 and Elitism and Genetic operators weights to 0, turning the other highlighted parameters untouched.

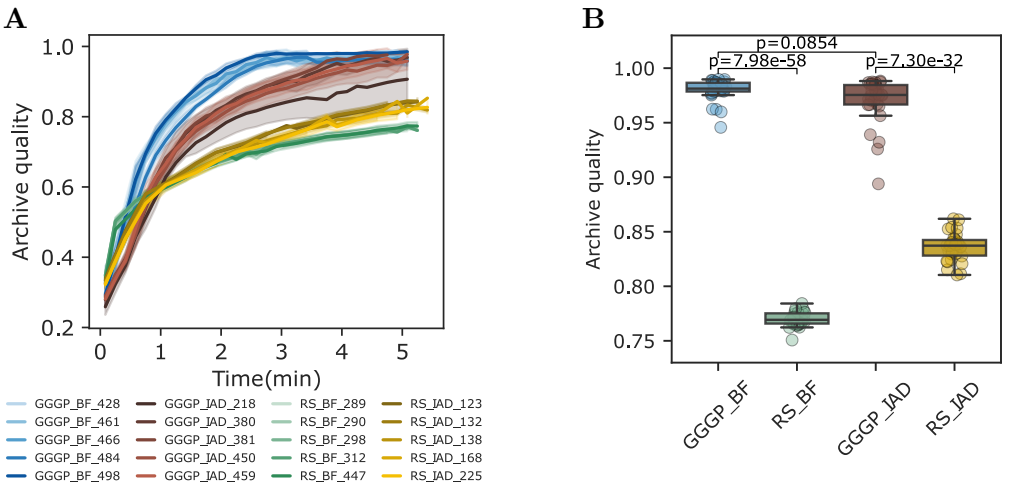

Figure 6.3: Archive quality evaluation between the GGGP and baseline experiments. **A** - Average archive quality throughout the search procedure for four strategies (GGGP_BinFiller, RandomSearch_BinFiller, GGGP_IAD and RandomSearch_IAD). We show only the top 5 trials of each strategy, each representing the average of 5 seeds. **B** - Distribution of the archive quality of the top trial of each strategy over 30 seeds. Statistical significance was assessed using Welch's t-tests for three pairs of samples: two comparing the means of the GGGP and baseline when using the same fitness function, and one comparing the GGGP with different fitness functions. P-values were adjusted for multiple testing using the Bonferroni correction.

Next, we executed 30 seeds of the top trial of each strategy (Table 6.2). The larger number of seeds confirms the effectiveness of GGGP with Bin Filler and highlights significant performance differences between GGGP and Random Search (Figure 6.3B). When comparing the two fitness functions, GGGP with IAD was competitive against Bin Filler, achieving a median archive quality of over 0.95. Interestingly, the hyperparameter search yielded a lack of novelty (weight

0 in Table 6.2), rendering the evolution highly dependent on the search space covered during population initialization.

To further explore the impact of the fitness functions, we conducted a small experiment wherein we used the best parameters derived by Optuna for GGGP with IAD and BinFiller, but exchanged the fitness functions between them. The best results stemmed from the parameter configuration derived for BinFiller, rather than from the specific fitness function used (Figure 6.4A). This suggests that the fitness function may not be the primary driver of the evolutionary search outcome; instead, the parameter setup appears to play a major role. Interestingly, when examining the average edit distances to the original sequence, we observed that the parameter setup obtained for IAD (IAD, BF_IAD experiments) yielded simpler genotypes, regardless of the fitness function (Figure 6.4B). Hence, although not being the most performant in archive quality, this configuration could prove beneficial for downstream interpretability applications.

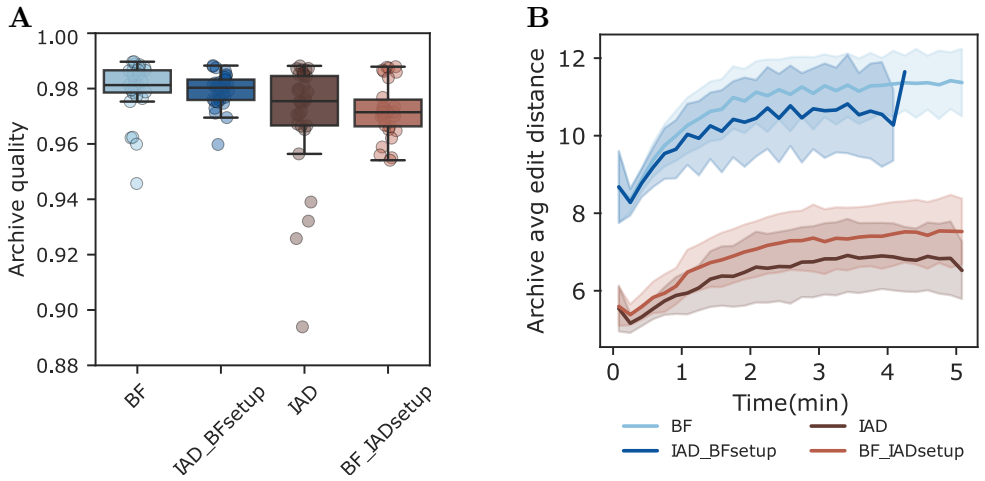

Figure 6.4: Evaluation of the fitness function effect in the evolutionary search outcome. Four different strategies are displayed: Optuna-derived GGGP parameters for BinFiller and IAD, plus two experiments. These involve using IAD as fitness function with the BinFiller-optimized parameters (IAD_BFsetup), and vice versa (BF_IADsetup).**A** - Boxplots displaying archive quality variability based on 30 different seeds. **B** - Lineplots showing the average edit distance (number of nucleotides that differ from the original sequence) of the archive sequences along the evolution time based on 30 different seeds.

6.6 Ablation studies reveal domain-specific insights

Using the best configuration, GGGP with Bin Filler, we explored how different components can effect evolution performance.

6.6.1 Lexicase selection

First, we replaced Tournament selection with Lexicase selection [261] with two objectives. Besides maximizing the quality of the archive (first objective), we also minimized the edit distance between the generated sequences and the original one. This second objective aims to reduce the syntactic diversity of the generated dataset, thereby potentially enhancing its explainability.

Surprisingly, Lexicase selection performed very poorly regarding archive quality (Figure 6.5A). While the second objective helped the generation of simpler genotypes (lower average edit distance of the archives compared to the best configuration), this hurt overall performance. Custom hyperparameter tuning for Lexicase selection did not improve the results (data not shown). In addition, the slow rate at which sequences were added to the archive suggests that Lexicase hindered the exploration of sequence space (Figure 6.5B). This is likely because most sequences added to the archive were selected based on the primary objective. Conversely, the second objective, which probably favored individuals with a single SNV, primarily slowed down the evolutionary process. This observation reflects biological complexity: higher edit distances, which are important for exploring epistatic interactions, are likely necessary for fully capturing the biological fitness landscape.

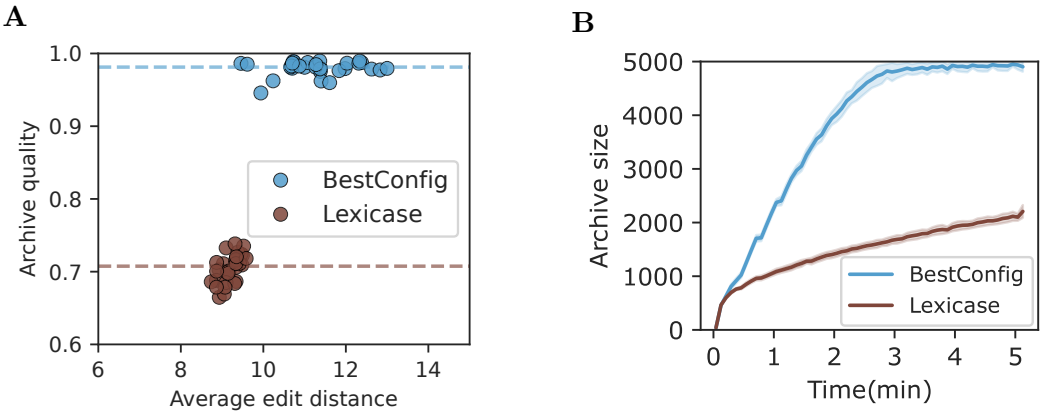

Figure 6.5: Impact of Lexicase selection in the final archive. **A** - Average edit distance as a function of archive quality for tournament and lexicase selection across 30 different runs. The horizontal lines reflect the median archive quality of each selection method. **B** - Averaged archive size throughout 30 different runs.

6.6.2 Custom mutation operator

Next, we examined how the custom mutation operator affects archive quality. The best setup displayed a custom mutation rate of 0.8, where 80% of GGGP mutations swap `DiffUnits` with others in spatial proximity. We conducted experiments testing ten additional rates by

incrementally decreasing this value from 1 to 0, consequently increasing the standard random mutation rate.

We found that reducing the custom mutation rate negatively impacted the overall quality of the archive, especially when relying solely on standard GGGP random mutation (rate 0), as illustrated in Figure 6.6. This outcome strongly suggests that exploring `DiffUnit`s in a more localized manner is targeting functional motifs faster than when mutating across the whole sequence. These results highlight the benefit of embedding domain and problem-specific properties in the design of the evolutionary algorithm.

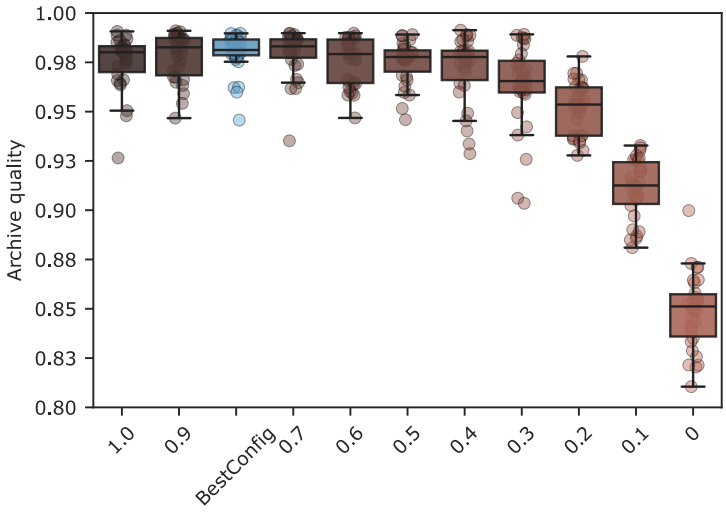

Figure 6.6: Impact of the frequency of the custom mutation operator (vs standard GP tree-based mutation) in the final archive quality, across 30 seeds.

6.6.3 Restricting the types of perturbations

Lastly, we assessed the performance impact of omitting certain `DiffUnits` from the grammar. As expected, when omitting both deletions and insertions (only SNVs are allowed) the archive quality was reduced, as there was no sufficient sequence edits to explore the whole model prediction landscape (Figure 6.7A, *NoDelsAndIns* points).

Interestingly, when only insertions were excluded from the evolution, archive quality also dropped to similar levels as for the *NoDelsAndIns* configuration. In contrast, the exclusion of deletions (*NoDels*) appeared to have minimal impact on performance (Figure 6.7A). This pattern suggests that insertions play a crucial role in improving black box prediction coverage. In particular, this is evident for score bins greater than 0.9, which are the hardest to reach under all conditions: grammars without insertions (*NoIns, NoDelsAndIns*) contribute disproportionately fewer sequences at these bins compared to grammars with insertions (*NoDels, BestConfig*, as shown in Figure 6.7B).

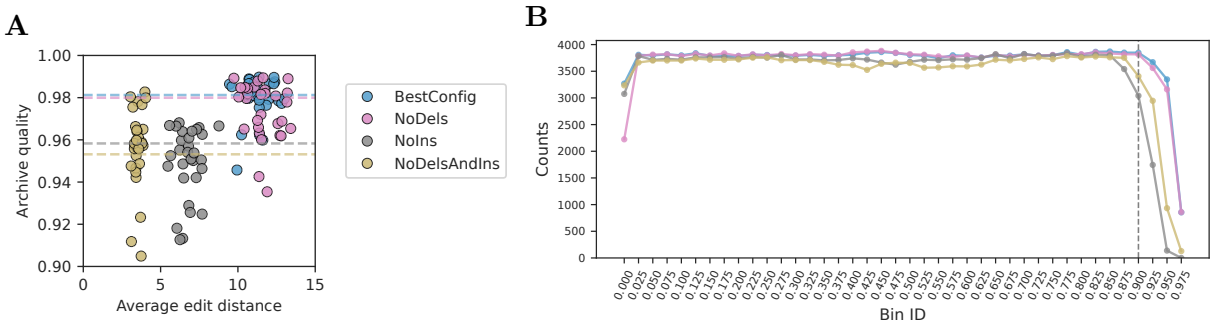

Figure 6.7: Impact of excluding specific grammar nodes on archive quality. Four conditions were tested: the best configuration, the best configuration without deletions (*NoDels*), without insertions (*NoIns*) and without both (*NoDelsAndIns*). **A** - Average edit distance and archive quality for each condition across 30 seeds. Horizontal lines represent the median archive quality for each condition. **B** - Number of unique sequences generated across 30 seeds in each score bin for each condition.

6.7 Generalizing to other input sequences reinforces differences against baseline

To assess the generalizability of our approach to other sequences, we used RNA-Seq data from the ENCODE [113], as done in Chapter 5. Specifically, we focused on the gene knockdown of the RBFOX2 gene, a known regulator of alternative splicing [262–264], and extracted the list of differentially spliced exons upon knockdown to be subjected to analysis using our GGGP approach the Random Search baseline. We applied slightly different filters compared to Section 5.1.1, where we did not exclusively focus on splicing events in protein-coding genes, resulting in 144 exons to study. We have also integrated SQUID [34] into the benchmarks as a second baseline (see Appendix C.1 for details on how it was done).

We found that GGGP outperformed Random Search and SQUID in dataset generation quality across a diverse set of input sequences (Figure 6.8). On average, GGGP achieved 0.93 in archive quality, compared to Random Search's 0.63 and SQUID's 0.60, representing a ≈30% improvement. Random Search outperformed GGGP in only 3 out of 144 sequences (0.02%), and even then, the margin was minimal. In addition, we aimed to assess the impact of the sequence length on performance. Since the proof-of-concept was carried out on a relatively short sequence (FAS exon triplet plus 100 nucleotides on each side, totaling 1743 nucleotides), we examined how the generated archives stand up to much longer sequences, which reflect larger search spaces. We found more pronounced differences in longer sequences, underscoring that our approach better navigates larger search spaces (Figure 6.8, `GGGP-RS` and `GGGP-SQUID` heatmap annotations). Notably, GGGP performance appears unaffected by sequence length, maintaining consistent archive quality across different sequence sizes (Table 6.3).

We further illustrate the influence of the original sequence prediction on the search outcome.

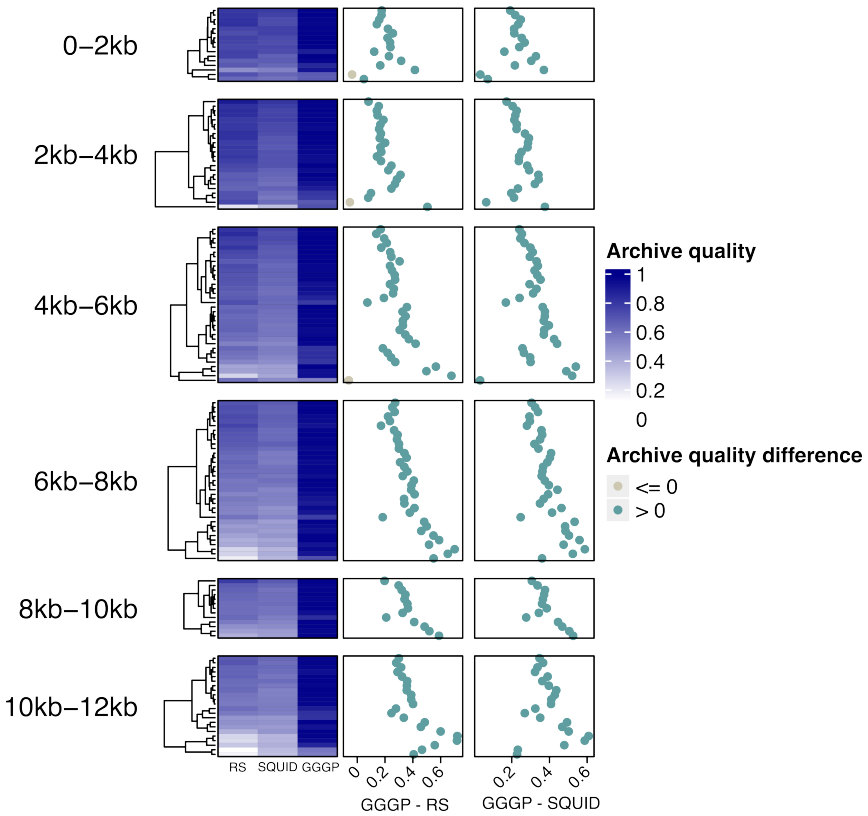

Figure 6.8: Archive quality comparison across sequences of varying lengths (binned vertically), focusing on performance differences between GGGP and the baselines. The heatmap annotations, in green, show the performance difference between GGGP and each of the baselines (Random Search and SQUID).

Table 6.3: Mean archive quality for sequences at different sequence length intervals.

Sequence length class	Number of sequences	Random Search	SQUID	GGGP
0-2kb	16	0.732	0.702	0.92
2kb-4kb	24	0.743	0.678	0.927
4kb-6kb	34	0.652	0.603	0.93
6kb-8kb	35	0.571	0.552	0.946
8kb-10kb	13	0.596	0.573	0.963
10kb-12kb	22	0.501	0.514	0.911

In particular, when the model predicted values close to 0 or 1, we observed larger performance differences between our approach and the baselines, especially when the original exon was predicted with high probability (Figure 6.9). This makes biologically sense because exons predicted to be constitutive (close to 1) or barely included (close to 0) in the final RNA transcript are inherently resistant to changes across the PSI landscape [265]. These findings underscore

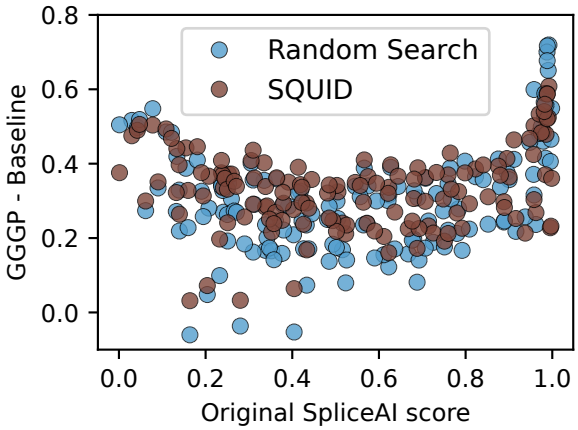

Figure 6.9: Differences in archive quality between GGGP and the baselines (Random Search and SQUID) as a function of the SpliceAI score of the original sequence.

the effectiveness of our approach, especially considering our deliberate avoidance of perturbing highly sensitive regions in the sequences, such as splice sites.

6.7.1 Motif analysis of synthetic datasets

We explored which known biological motifs were captured by the generated datasets by quantifying the frequency of what we term "motif disruption" events - motif gains/losses relative to the original sequence - in datasets generated by GGGP and Random Search. To do so, we performed a classical motif analysis on the generated datasets to assess the presence of ground-truth motifs in the synthetic sequences. We scanned sequences with motif PWMs using FIMO [215] from the MEME suite v5.5.3. We used oRNAment [217] as the motif database, and selected RBPs associated with RNA splicing regulation (according to [113]), which resulted in PWMs from 36 RBPs for scanning. Only motif matches with a p-value < 0.0001 were considered.

From the motif scanning step, we additionally processed the output to measure the frequency of motif disruption events per dataset, defined as the total number of motifs gained or lost relative to the original wild-type sequence, divided by the dataset size. The dataset size is just a normalization factor, since the number of generated sequences was not consistent between Random Search and GGGP-based datasets.

Our analysis revealed a higher proportion of motif disruption events in GGGP datasets (Figure 6.10, `Motif Diff` heatmap annotation), indicating that sequences generated by GGGP contain richer biological information. However, 32 exons (22.2%) had datasets with a lower number of disruption events (relative to the dataset size) compared to those generated by Random Search. As GGGP datasets effectively cover the prediction landscape, it remains unclear whether the model has learned previously unknown motif syntax or if the perturbations

influencing the model's predictions are merely spurious artifacts.

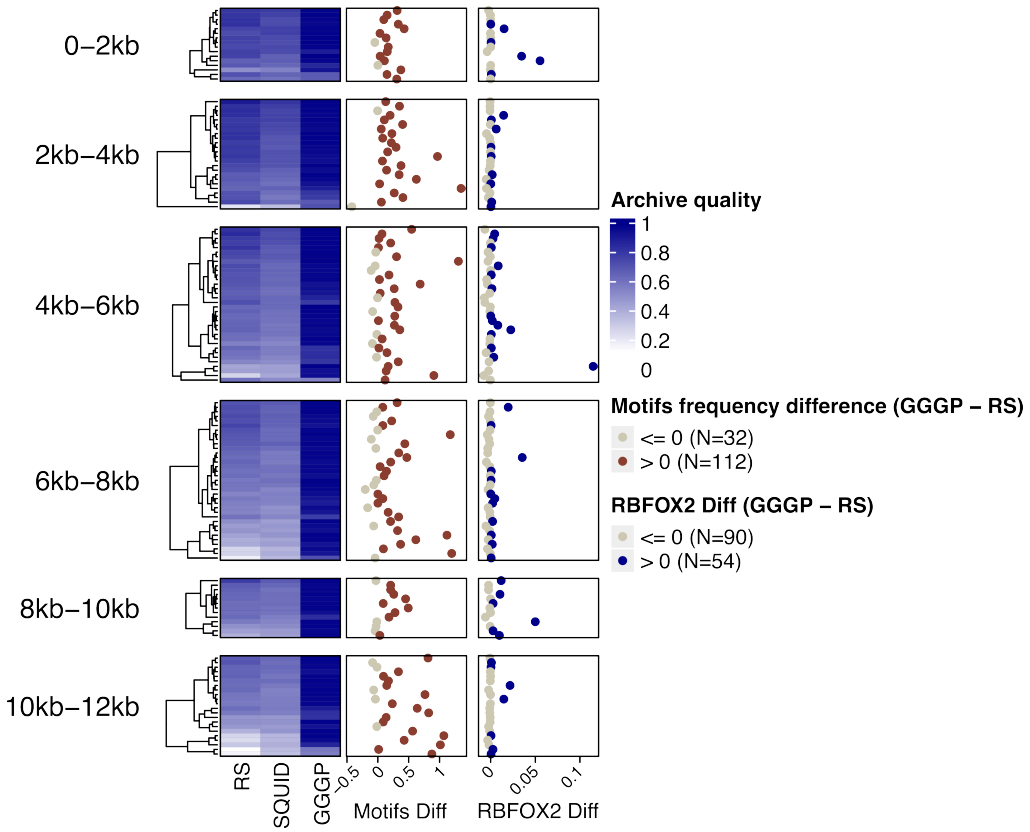

Figure 6.10: Archive comparison across sequences of varying lengths (binned vertically), focusing on motif disruption events. Each dataset consists of the same run performed with five different seeds. The heatmap annotations show the difference in the relative frequency of motif disruptions between GGGP and Random Search for all motifs in the database (`Motifs Diff`) and for RBFOX2 only (`RBFOX2 Diff`).

We additionally explored motif disruption events for RBFOX2, specifically. Since the analyzed sequences displayed splicing changes upon RBFOX2 knockdown, we wondered whether the generated datasets, which cover the model prediction landscape, would also capture RBFOX2 motifs being gained or lost in the synthetic sequences. Interestingly, we observed no enrichment of RBFOX2 motifs in GGGP datasets compared to Random Search, except for a couple of exceptions (Figure 6.10, `RBFOX2 Diff` heatmap annotation). Upon further examination of the results, we observed that 75 out of 144 original sequences (52%) did not exhibit any RBFOX2 motif hits, and only 4 sequences contained more than 3 RBFOX2 motifs. The low frequency of RBFOX2 motifs in real (wildtype) sequences suggests that many alternative splicing events detected may likely be attributed to indirect effects of the RBFOX2 knockdown (e.g., involvement as a member of a protein complex) rather than direct binding to RNA. Another possible explanation for the similar frequency of RBFOX2 disruption events in GGGP

and RS is the contextual landscape in which these motifs were gained or lost (e.g., their sequence location or the presence of other perturbations in the same sequence). Although the frequencies of RBFOX2 disruption events were similar, sequences with these events displayed distinct SpliceAI delta score distribution profiles in the GGGP and Random Search datasets (Figure 6.11). Synthetic sequences generated with GGGP were semantically richer, while Random Search-derived sequences had RBFOX2 motif disruptions that did not affect the SpliceAI score, making them irrelevant for the model's predictions. This suggests that GGGP may leverage RBFOX2-related perturbations, alone or in combination with other perturbations, to drive predictions towards diverse regions of the prediction landscape.

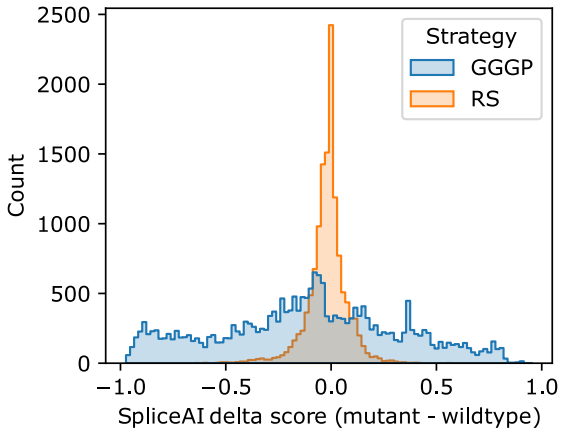

Figure 6.11: Distribution of SpliceAI delta scores for synthetic sequences with RBFOX2 motif gains and losses compared to the original sequence.

6.8 Studying the impact of an alternative PWM grammar

So far, we have proposed a GP approach that constrains possible perturbations using a grammar. The grammar, described in Section 6.3.1, encodes a set of productions that define the rules for generating synthetic sequences. Combined with Meta-Handlers, which enhance the grammar's expressive power by imposing type refinements (e.g., avoiding perturbations of splice site regions), we have successfully generated synthetic datasets that cover the prediction landscape of SpliceAI. However, the grammar remains relatively unconstrained since perturbations are random. This may result in many ineffective perturbations and consequently longer convergence times before achieving biologically meaningful changes.

Hence, we designed another grammar that restricts the perturbations to a set of predefined motifs from PWMs (Figure 6.12). Just like the previous grammar (Figure 6.2), the PWM grammar defines individual genotypes as a sequence of 1 to 6 perturbations.

A

Perturbation Sequence	DiffSeq	::=	DiffUnit{1-6}
Perturbation	DiffUnit	::=	MotifSNV
		|	MotifIns
		|	MotifSubst
		|	MotifDel
		|	MotifAblt
SNV at Motif	MotifSNV	::=	SNV(MotifPos, Elem, Nuc)
Motif Insertion	MotifIns	::=	Ins(SeqPos, MotifSeq)
Motif Substitution	MotifSubst	::=	Subst(SeqPos, MotifSeq)
Motif Deletion	MotifDel	::=	Del(MotifPos)
Motif Ablation	MotifAblt	::=	Ablt(MotifPos, RandomNucs)
Motif Sequence	MotifSeq	::=	RBP{1-N}
RNA-binding Protein	RBP	::=	Motif{1-M}
Motif sequence	Motif	::=	Nuc+
Random Nucleotides	RandomNucs	::=	Nuc+
Motif Position	MotifPos	::=	int
Sequence Position	SeqPos	::=	int
Element within motif	Elem	::=	int
Nucleotide	Nuc	::=	A
		|	C
		|	G
		|	T

B

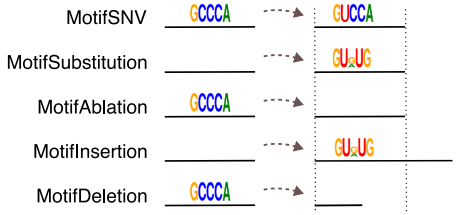

Figure 6.12: Perturbation grammar from PWMs. **A** - Core structure of the grammar used to represent an individual in respect to the original sequence, presented in EBNF. Underlined symbols are terminals or meta-handlers. Node names are shortened for space constraints. **B** - Types of grammar nodes (`DiffUnits`) encoded in the grammar.

Now, each perturbation can be one of five types: `MotifSNV(MotifPos, Elem, Nuc)` employs an SNV at a given element within an existing motif, `MotifSubstitution(SeqPos, MotifSeq)` replaces a portion of the sequence with a motif at a given position, `MotifAblation(MotifPos, RandomNucs)` removes a motif by shuffling out that portion of the sequence, `MotifInsertion(SeqPos, MotifSeq)` inserts a motif at a given position, and `MotifDeletion(MotifPos)` deletes a motif from the sequence. Of note, `MotifSubstitution` and `MotifAblation` ensure the same sequence length after replacement, whereas `MotifInsertion` and `MotifDeletion` result in a sequence length change. A visual representation of the grammar nodes available is show in Figure 6.12B.

Importantly, this grammar requires an initial motif scan at the beginning of the evolutionary run to extract the motif locations in the original sequence. Accordingly, we developed a set of custom Meta-handlers to restrict the perturbation space to these existing motif locations. We also maintained the rationale of avoiding perturbations around splice sites of the exon triplet (`[-10, 2]` and `[-3, 6]` around splicing acceptors and donors, respectively). We hypothesized that by imposing more fine-grained biological constraints (motif vs. random perturbations), we could achieve equally good performance with faster convergence, despite the increased restrictions on the search space (less available regions to perturb).

By comparing the performance of GGGP against the Random Search baseline using both the original and PWM grammars, we found that regardless of the grammar, GGGP outperformed

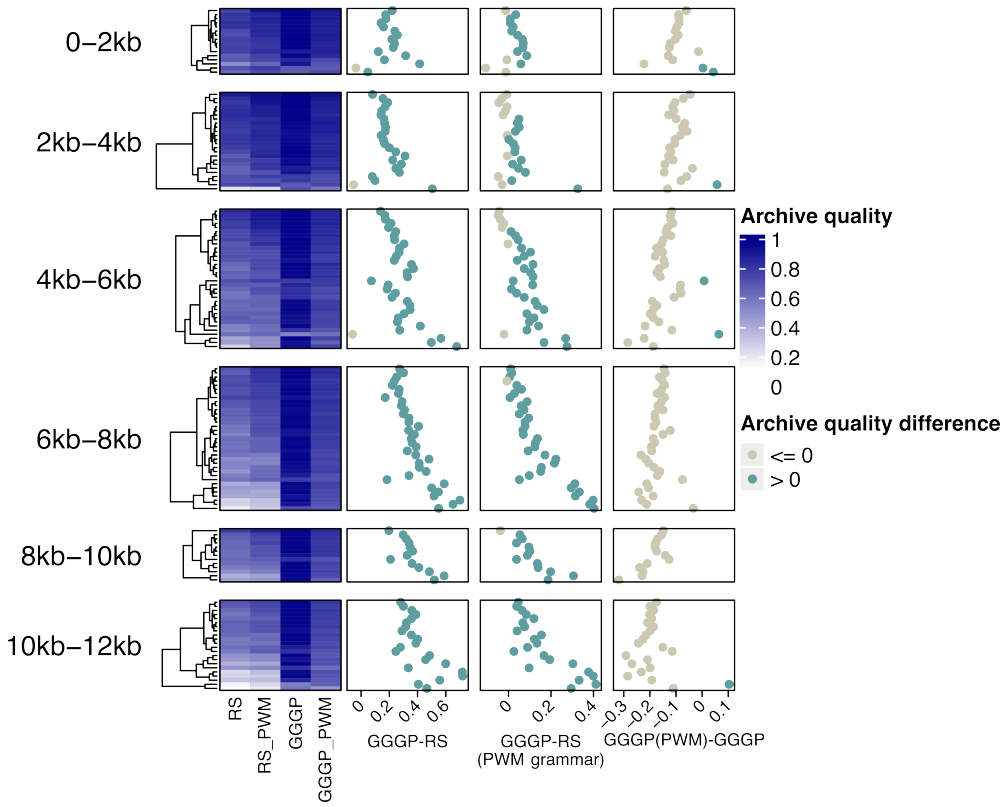

Figure 6.13: Archive quality comparison across sequences of varying lengths, focusing on a grammar encoding perturbations from PWMs. The first two heatmap annotations show the performance difference between GGGP and the corresponding baseline, and the right-most annotation (`GGGP(PWM) - GGGP`) compares GGGP approaches using different grammars.

Random Search in the same order of magnitude (Figure 6.13 `GGGP-RS` and `GGGP -RS (PWM grammar)` heatmap annotation). However, contrary to our hypothesis, GGGP using different grammars exhibited a large performance difference, with the original grammar producing substantially better results (Figure 6.13, `GGGP(PWM) - GGGP` heatmap annotation). These findings reveal two important insights. First, the impact of the grammar on the evolutionary search outcome is substantial. Second, the PWM grammar may be too restrictive, both at the locations where perturbations can occur (e.g., ablations and deletions are limited to known motif locations) and the nucleotides that can be inserted or substituted (restricted to known motif sequences). Because the deep learning model has learned unbiased motif representations from the training data, the original grammar, which allows for random combinations of nucleotides, was more effective in exploring the prediction landscape of the model. Nevertheless, this grammar can be very useful for other type of experiments, as we will see in the next chapter.

6.9 Exploring Pangolin as the deep learning oracle

In this section, we explore the impact of the oracle in the evolutionary search process. Instead of SpliceAI, we employed Pangolin [141], which predicts splicing in four different tissues. By default, Pangolin averages predictions across all tissues. However, due to the additional computational overhead of running inferences for each tissue, we focused on a single tissue (heart) to render a fair comparison with SpliceAI. This approach ensures that the evaluation of the population at a given generation takes approximately the same amount of time. We used the same RBFOX2-sensitive sequences and the grammar encoding random perturbations, as it previously yielded the best results (Figure 6.13).

The datasets generated by Pangolin displayed lower quality than those generated by SpliceAI, with a median archive quality of 0.683 compared to 0.93 for SpliceAI (Figure 6.14). Pangolin's derived datasets were of better quality in only 5 out of 144 sequences. These results do not necessarily reflect the quality of the oracle itself. Instead, they suggest that it is more challenging to cover the prediction landscape of Pangolin within the same time budget as SpliceAI. In other words, SpliceAI is more sensitive to perturbations in the input sequence. Let us discuss this in more detail.

Pangolin was trained using splicing quantifications from RNA-Seq data, representing an *in vivo* snapshot of the splicing landscape. Unlike SpliceAI's binary nature (is a position a splice site or not), Pangolin predicts continuous values of splice site usage across different tissues. This task likely required Pangolin to focus on more compact features to accurately predict continuous labels and their variations across tissues, making it less sensitive to random perturbations. As an example, if an exon is predicted with a very low score in heart tissue, and similar sequences were observed in the training data at very low inclusion levels, it is reasonable to speculate that no perturbations will easily lead splice site usage up to 1, assuming no data distribution shifts in the synthetic sequences, which we explicitly control for by imposing a low number of perturbations per sequence in the grammar. Hence, the lower quality observed in archives guided by Pangolin (lower diversity in the model's semantic space) may actually be a more realistic representation of biology, considering the cell-type-specific nature of splicing regulation.

As before, we analyzed the motif disruption events and compared their prevalence between datasets generated with Pangolin and SpliceAI. We found a similar proportion of motif disruption events between the models (Figure 6.14, `Motif Diff` heatmap annotation), suggesting that known motifs affect the models' predictions at a similar level. This observation also holds for RBFOX2 motif gains or losses (Figure 6.14, `RBFOX2 Diff` heatmap annotation).

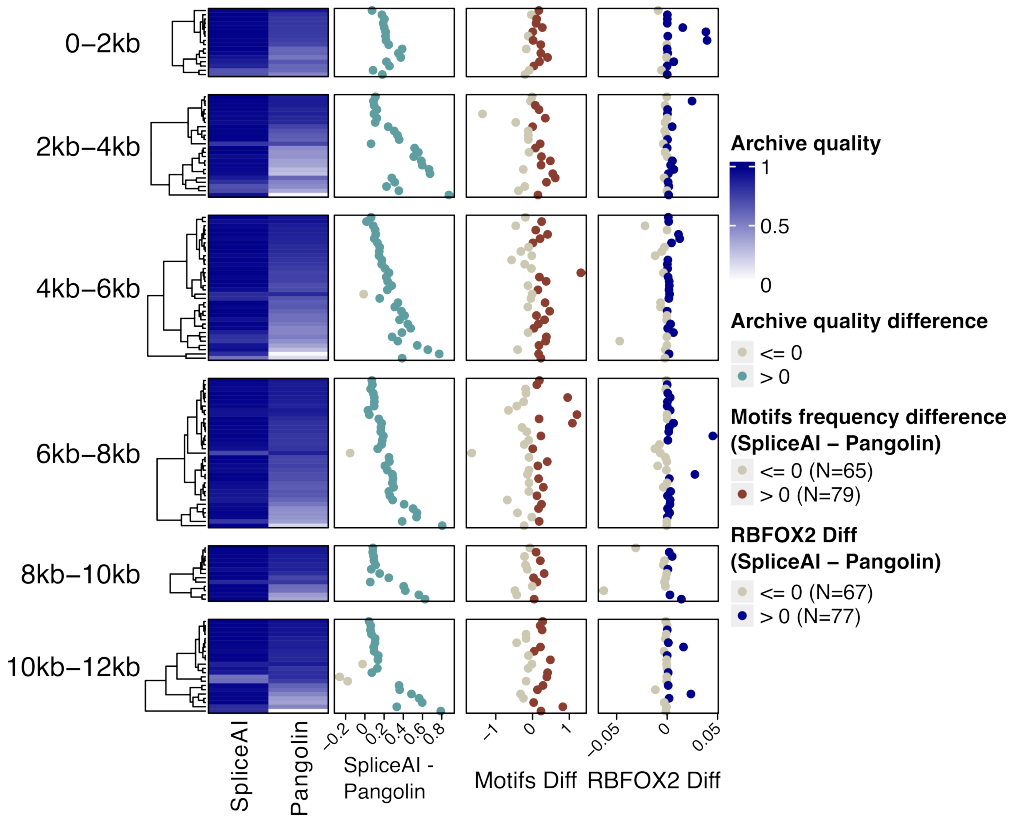

Figure 6.14: Archive quality comparison across sequences of varying lengths, focusing on two different oracles, SpliceAI and Pangolin, using the grammar encoding random perturbations. The first heatmap annotation shows the archive quality differences between SpliceAI and Pangolin. The two right-most annotations show the differences in the relative frequency of motif disruptions between SpliceAI and Pangolin's derived datasets for all motifs in the database (`Motifs Diff`) and for RBFOX2 only (`RBFOX2 Diff`).

6.10 Conclusion

Based on the accumulated evidence, we conclude that our grammar-guided Genetic Programming approach greatly improves over random sampling on the task of generating semantically meaningful local synthetic datasets. We found this to be true not only in a relatively short, controlled sequence (Section 6.5) but also for 144 sequences that are diverse in their length, genomic location, and original black box score (Section 6.7). In these sequences, our approach achieved an average 30% improvement over the baseline.

Our results have also highlighted the advantage of introducing domain knowledge in the problem specification. By constraining highly sensitive regions in the sequence from being explored, we force the evolutionary algorithm to learn alternative yet biologically interesting paths to achieve semantic diversity. Indeed, we proved that synthetic sequences generated by GGGP contain richer biological information (motif analysis in Section 6.7.1). Furthermore, our

custom mutation operator that promotes locality proved beneficial compared to GGGP's default tree-based mutation (Section 6.6.2).

Finally, we demonstrated the impact of both the grammar and the model on the quality of the generated datasets. Our results indicated that a simpler, less constrained grammar more effectively covered the model's prediction landscape (Section 6.8). Regarding the model, SpliceAI-generated datasets exhibited higher quality than those generated by Pangolin, implying that SpliceAI is more sensitive to perturbations and that a greater number of sequence positions contribute to its predictions, which is not necessarily advantageous for its interpretability (Section 6.9).

Chapter 7

DRESS: a flexible framework for splicing interrogations guided by deep learning models

In this chapter, we build upon the concepts introduced in Chapter Chapter 6 by presenting a software called Deep learning based Resource for Exploring Splicing Signatures (DRESS), designed with a user-friendly client interface. While not introducing new scientific findings, this short chapter demonstrates a practical application of grammars for computational biologists interested in RNA splicing research. As DRESS is actively under development, we also highlight features planned for future releases (Figure 7.1).

The main goal of this software is to provide a flexible framework for RNA splicing research guided by performant deep learning models. Its strength lies in the embedded domain knowledge. To illustrate its capabilities, we explore use cases involving exon 18 of the *MYOM1* gene, whose alternative splicing is critical for cardiac development [266, 267].

7.1 Dataset Generation

`generate` is the main command of this framework. It reads an input sequence and generates synthetic sequences by performing data augmentations based on genetic algorithms and a deep learning oracle. The augmentations are controlled via grammars, which encode constraints on the sequence space. An overview of the main algorithm steps is presented in Algorithm 1.

The primary goal of the genetic search is to evolve sets of perturbations that span the prediction landscape of the deep learning model. The command is built on top of GeneticEngine [39], a framework for Genetic Programming in Python.

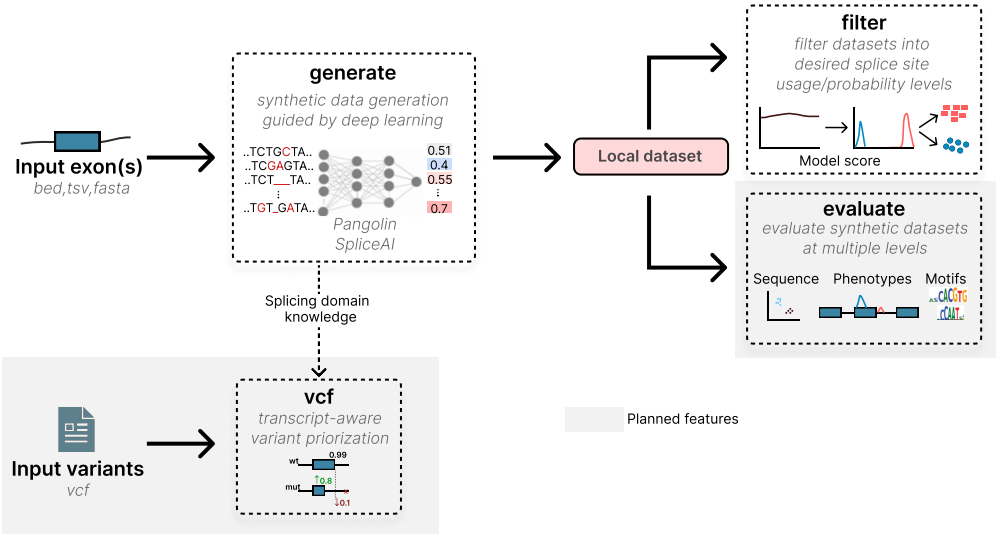

Figure 7.1: Schematic of DRESS framework.

Algorithm 1: Genetic Programming Algorithm for Synthetic Sequence Generation

Input: Original sequence S, Perturbation grammar G, Fitness function F, Deep learning oracle O, Fitness Threshold T_f, Stopping criteria C, Number of archive bins b, Max sequences per bin m, Elitism rate e, Novelty rate n, Selection rate s

Output: Semantically rich dataset D

Initialize population P with N random individuals based on S and G;

Initialize archive A with b bins, each with capacity m;

Start timer;

$P, A \leftarrow$ **EvaluatePopulationAndUpdateArchive**(P, O, A, F, T_f);

while *stopping criteria* C *not met* **do**

  Elitism: Select top $e \times N$ individuals based on fitness to carry over unchanged;

  Novelty: Generate $n \times N$ new individuals based on S and G;

  Selection: Perform selection on $s \times N$ individuals, apply crossover and mutation operators based on G and their specific rates;

  Update population P with new individuals;

  $P, A \leftarrow$ **EvaluatePopulationAndUpdateArchive**(P, O, A, F, T_f);

end

Return archive A as dataset D;

Function `EvaluatePopulationAndUpdateArchive`(P, O, A, F, T_f)**:**

  Correct population P to ensure all individuals are valid based on G

  Convert P phenotypes into sequences

  Run oracle O in parallel on all sequences in P to get predictions y

  foreach *individual* $i \in P$ **do**

    Calculate fitness F_i based on y_i and archive A, considering current bin capacities

  foreach *individual* $i \in P$ **do**

    if $F_i > T_f$ **then**

      Add i to archive A

  return P, A

7.1.1 Input preprocessing

The software simplifies the entire preprocessing of the input. It only requires the input exon(s) in BED or tabular format, and automatically extracts genomic intervals such as surrounding exons and splice site locations from an internal cache built from a GENCODE GTF. If transcript IDs are provided, the extracted full input sequence will be transcript-specific, enabling the study of even non-coding transcripts (e.g., NMD transcripts). By default, DRESS uses the highest-ranked transcript that contains the input exon(s) (based on transcript flags such as `MANE` or `CCDS`). DRESS can handle sequences of variable size without losing domain-specific information. During inference, predictions for the target exon, which may be located at variable positions across the sequences, are properly managed.

7.1.2 Evolutionary algorithm

In Algorithm 1 we described a high level overview of the genetic algorithm. Each step in the process offers customizable options common to genetic algorithms, such as setting population size, defining stopping criteria (e.g., number of oracle inferences, generations, elapsed time, archive size), and choosing the selection method (e.g., Tournament, Lexicase).

Users can also adjust various hyperparameters related to the evolution of the population, such as the weight given to elitism, novelty, and selection. These ratios can be dynamically adjusted. For example, to initially explore the search space of possible perturbations and then exploit the best solutions later on, the novelty/selection ratio can be set to decrease over generations. For a purely random search over the perturbation space, the novelty rate can be set to 1.0.

DRESS also supports archive pruning, which simplifies individual genotypes by removing irrelevant perturbations that do not affect the sequence score. Additionally, several logging options are available to track the evolution of the population or archive, enabling downstream analysis of how combinations of splicing perturbations evolved.

Lastly, the default fitness functions promote individuals such that the generated datasets cover the full model prediction landscape (e.g, archives with high diversity). However, it can be easily adapted with different fitness functions that guide the genetic algorithm into datasets with desired PSI distributions.

7.1.3 Domain constraints are seamlessly integrated

The main advantage of the software is its ability to handle splicing-specific constraints. It would be impractical to implement them in any other general-purpose genetic algorithm framework. The domain knowledge is encoded through the definition of grammars with Meta-Handlers, that restrict the search space to domain-meaningful values. DRESS has native support for two grammars, the random perturbation grammar and the PWM grammar, introduced in the

previous chapter. Defining custom grammars is straightforward, as they are implemented as Python classes and Meta-Handlers are type refinements defined as Python Type annotations (e.g., an integer between 5 and 10).

By default, DRESS avoids perturbing positions near splice sites to prevent drastic outcomes when perturbing those regions. However, users can control these ranges via additional input arguments. Furthermore, domain-informative tags can be provided to restrict the search space further. For instance, to play with splicing without interfering with the cassette exon, one can run the software with `--untouched_regions cassette_exon`.

Let us explore these features in more detail. We use as input sequence the exon 18 of the *MYOM1* gene, which is predicted by Pangolin with an inclusion level of 0.3263 in heart tissue (average of the acceptor and donor positions). The donor site is predicted with low usage, 0.1007. We will then study the impact of unperturbed splice donor motifs on the splicing of a shuffled version of this sequence. To do this, we shuffle the input sequence of exon 18, along with its surrounding introns and exons, using the `--shuffle_input` option. The splice site regions will remain unchanged, controlled by the `--acceptor_untouched_range` and `--donor_untouched_range` options.

While testing varying ranges of unperturbed splice donors, we will keep the acceptor site fixed and unshuffled. This approach will help us determine how much of the real sequence context at the splice site borders is needed to activate splice donor site usage. Below is an example of the command used:

```
# We keep the acceptor motif fixed and unshuffled (last 10 nt of the intron
# and first 2 nt of the exon) while varying the unperturbed donor motif
# (X nt upstream and Y nt downstream of the donor site)
dress generate --model pangolin \
               --pangolin_tissue heart \
               --model_scoring_metric donor \
               --stopping_criterium n_generations \
               --stop_at_value 0 \ # No evolution is needed
               --shuffle_input shuffle \
               --acceptor_untouched_range -10 2 \
               --donor_untouched_range X Y
               myom1_exon18.tsv
```

The results show some interesting patterns (Figure 7.2). As a control, we observed that when shuffling all the input, including the splice donor dinucleotide (0 and 1 on the x-axis), the model predicts that the position is not a donor, regardless of how many exonic positions are left unshuffled. As expected, when we increase the number of nucleotides left unshuffled, the splice donor usage increases as the model recognizes a genuine donor region.

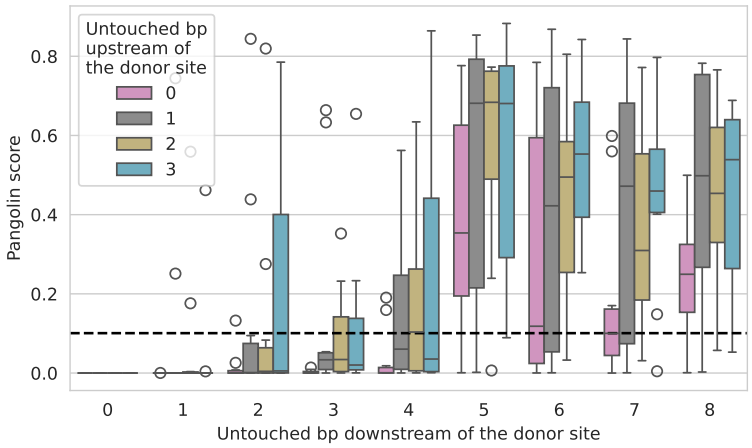

Figure 7.2: Splice donor usage predicted in heart tissue by Pangolin across shuffled *MYOM1* sequences with varying ranges of unperturbed donor motifs. Each boxplot represents the distribution of predictions across 10 shuffles. The x-axis values indicate the number of intronic positions (starting at the donor site) that remained unperturbed, preserving the real sequence context. The color of each group of boxplots represents the number of upstream positions (within the exon) that were left unperturbed. The dashed black horizontal line represents the unshuffled real sequence prediction.

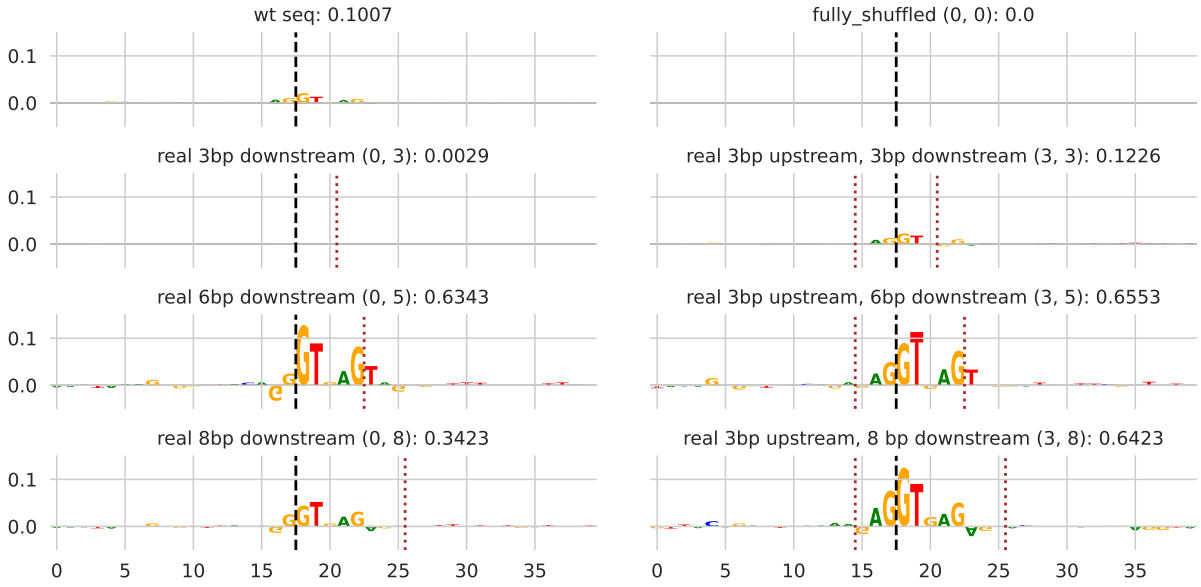

Figure 7.3: Feature attribution analysis for the splice donor position in the wildtype (real) sequence and in shuffled sequences with varying ranges of unperturbed donor motifs. DeepLIFTShap was used (from https://github.com/jmschrei/tangermeme/) to compute attributions using 20 reference samples. Each subplot title contains information about the unperturbed donor ranges as well as the Pangolin prediction. The black dashed line represents the exon/intron boundary, and the brown lines represent the unshuffled sequence stretches.

However, it is interesting to note that with 5 intronic positions left unperturbed, the model predicts higher donor usage compared to the wildtype sequence, even though the remaining genomic context is shuffled (Figure 7.2). Feature attribution analysis reveals higher attribution values (difference to reference samples) at the donor region for sequences with larger prediction scores (Figure 7.3). Because the splice donor motif remains unchanged, these results suggest that regions of the shuffled sequence contribute not only to higher Pangolin predictions but also to providing the proper context for the donor region to have larger attribution scores. The shuffling likely disrupts silencing elements that normally constrain donor usage to low levels in the real sequence. The alternative explanation - that shuffling created enhancing motifs - is less likely, as the higher scores were consistently observed across multiple shuffles (Figure 7.2).

We now perform another simple experiment by performing several evolutions restricting the regions of the exon triplet that can be perturbed (`--untouched_regions option`). In particular, we will search on each region individually to assess which regions are more informative to navigate through the PSI landscape of the cassette exon. Experiments are performed in two different tissues, tissue and brain. An example of the command used to search just in the cassette exon, avoiding perturbations in the flanking regions, is shown below:

```
# Example command restricting the search space to the cassette exon
dress generate --model pangolin \
               --pangolin_tissue heart \
               --untouched_regions exon_upstream \
                                   intron_upstream \
                                   intron_downstream \
                                   exon_downstream \
               --stopping_criterium archive_size time \
               --stop_at_value 5000 5 \ # Stops when any condition is met
               myom1_exon18.tsv
```

We observed varying levels of dataset quality depending on the region perturbed (Figure 7.4). Unsurprisingly, the cassette exon, which is relatively long with 288 bp, is the most sensitive region to variations in splice site usage, with datasets fully covering the model prediction space. Since splice site regions are not perturbed by default, these results suggest that dress effectively explores the exonic regulatory sequences to enhance or suppress splicing.

The quality of the datasets dropped considerably for other regions, with the upstream intron showing more importance than the downstream intron, likely due to the regulation of the branchpoint signal. The upstream and downstream exons contribute poorly to the splicing of the cassette exon. Interestingly, there appears to be some tissue specificity, as datasets are generally of higher quality for the brain. This may indicate that the splicing of this exon is more

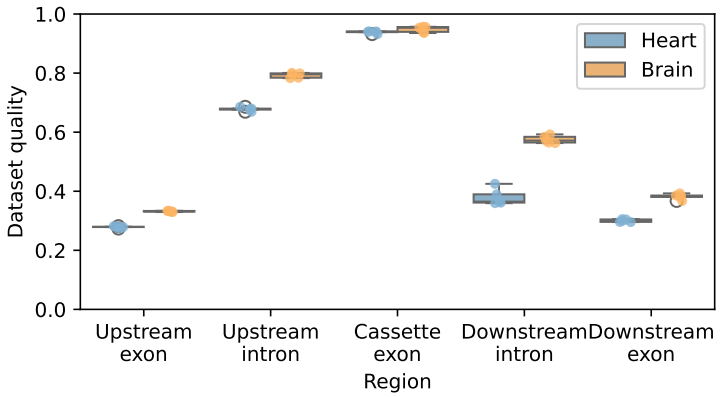

Figure 7.4: Quality of the generated datasets (measured as in Equation (6.1)) by constraining the search space to individual regions of the exon triplet across two tissues.

tightly regulated in heart tissue, which is consistent with the importance of *MYOM1* exon 18 in cardiac development.

7.1.4 Individual sequence explainability

Each individual, representing a list of perturbations on the original sequence within the population, is encoded using a grammar. The design of this grammar can make the phenotype highly expressive and informative. By embedding valuable information in the individual's phenotype, we enable direct assessment of certain components in the model's prediction. This includes the location of the perturbation, distances to splice sites, and introduced or removed motifs.

For instance, consider the same *MYOM1* sequence we have been working with, with a wildtype prediction of 0.32 in the heart tissue. We then generate synthetic sequences and, after a few generations, examine the sequences that drive the prediction further from the wildtype. Below, we present an illustrative command and examples of resulting sequences both with the random perturbation grammar and the PWM grammar (using `--which_grammar motif_based`):

```
dress generate --model pangolin \
               --pangolin_tissue heart \
               --stopping_criterium time \
               --stop_at_value 2 \ # Let it run for 2 minutes
               --max_diff_units 2 \ # Max number of perturbations per seq
               --which_grammar random \
               myom1_exon18.tsv
```

Table 7.1: Examples of phenotypes for synthetic sequences generated with two different grammars.

Grammar	Phenotype	Prediction
Random	`Insertion[5016,TGCA,Exon_cassette,16]|` `Deletion[5256,5260,Exon_cassette,31]`	0.7095
Random	`SNV[4989,A,Intron_upstream,11]|` `Insertion[5137,AT,Exon_cassette,137]`	0.0307
PWM-based	`MotifInsertion[5281,TRA2A,8,Exon_cassette,6]|` `MotifInsertion[7690,SFPQ,9,Exon_downstream,2403]`	0.6415
PWM-based	`MotifInsertion[4976,RBM5,7,Intron_upstream,24]|` `MotifDeletion[5051,SF1,7,Exon_cassette,51]`	0.0177

As Table 7.1 shows, the phenotype of individual sequences is intrinsically interpretable. For example, with the random perturbation grammar, two perturbations in the cassette exon - one insertion and one deletion at 16 and 31 bp away from the splice acceptor - drive the Pangolin prediction up to 0.7095. In another example, using the PWM-based grammar, a TRA2A motif insertion of length 8 at the cassette exon, 6 bp upstream of the splice donor, along with the insertion of a SFPQ motif of 9 bp at the exon downstream, enhances exon inclusion to 0.6415. The expressive power of individual phenotypes allows for straightforward exploration of 'what-if' scenarios at the sequence level, providing valuable counterfactuals to study the model's behavior.

7.2 Dataset filtering

The `filter` module is useful for filtering datasets created with the `generate` command. It allows users to filter sequences based on their predicted PSI or splice site probability values, or on specified tags. By default, the dataset generation process aims to cover the entire prediction landscape. However, using this module the user can create subsets of data with desired splicing outcomes.

For example, if we want to train a decision tree to classify sequences with varying PSI levels - let's say a delta PSI change of 0.5 relative to the reference - we would filter our *MYOM1*-exon18 synthetic dataset into two classes: sequences predicted at wildtype values (≈0.32) and those predicted at ≈0.82. This can be achieved by running the following commands, where the `--allowed_variability` option controls the deviation from the target value allowed:

```
dress filter --dataset myom1_dataset.csv.gz \
             --target_dpsi 0 \
             --allowed_variability 0.05 \
             --outbasename wt_seqs
```

```
dress filter --dataset myom1_dataset.csv.gz \
             --target_dpsi 0.5 \
             --allowed_variability 0.05 \
             --outbasename dpsi0.5_seqs
```

Alternatively, if we want to extract all sequences with increased inclusion levels compared to the wildtype sequence by a given difference, we would run the following command:

```
dress filter --dataset myom1_dataset.csv.gz \
             --tag higher \
             --delta_score 0.4 \
             --outbasename high_psi
```

This command would output a dataset with sequences predicted to have PSI levels higher than 0.72. These are just two examples, but the module allows for additional combinations of filtering criteria, based both on the model's predictions and on specific tags.

7.3 Planned features

We are developing another module to evaluate the generated datasets at various levels. This module will include examining sequence motifs using classical motif analysis tools (e.g., FIMO) and attribution methods such as DeepLIFTShap. Additionally, it will provide insights by parsing and aggregating the phenotypes across the dataset, which can be used to highlight enriched perturbed regions of the sequence across the prediction landscape. This will aid in debugging and explaining the black box model. This module is not yet available, but we are actively working on it.

Another planned feature is the addition of the `vcf` module,which will enable users to score variants in VCF format. This feature will extend the scoring capabilities of SpliceAI or Pangolin standalone packages by integrating transcript-specific predictions, considering hypothetical transcript isoform switches (based on contrasting splice site usage/probability due to the variant), and allowing the testing of multiple variants simultaneously. This last property will be particularly useful when, for instance, a patient has multiple variants in the same gene in close proximity and the user wants to understand the combined effect of these variants on splicing.

7.4 Final remarks

We developed a software to study RNA splicing by treating a deep learning model as an oracle for experimental validation. Currently, DRESS supports SpliceAI and Pangolin, two of the most widely used splicing sequence-based models. By providing a simple client interface, we aim

for this software to be a valuable tool for splicing research. The insights it provides may help generate new hypotheses and design lab experiments for further validation.

However, it is important to note that running the main command, `generate`, is impractical in environments without GPU availability. Due to its reliance on model inferences during population evaluation, running it on a CPU would result in significantly slower execution.

DRESS is available at `https://github.com/PedroBarbosa/dress`, and we are working towards providing comprehensive documentation, tutorials, and additional use cases similar to those provided here.

Chapter 8

Discussion and conclusion

While each chapter includes its own discussion, this final chapter integrates the overall findings of this work into a broader context, particularly considering the unprecedented pace at which deep learning technologies are evolving. We also take an honest, critical view of possible limitations of our work with room for improvement.

8.1 Variant effect prediction is not a solved task

Variant effect prediction is a critical task in genomics, serving as a fundamental step in interpreting the molecular mechanisms underlying genetic diseases and traits. Improved mapping of genotype-phenotype relationships enhances our understanding of the basis of life and brings us closer to the promising, yet still distant, goal of personalized medicine.

Our research demonstrates that deep learning models can effectively predict the impact of genetic variants on splicing (Chapter 4). Importantly, this performance holds for variants disrupting splicing through diverse molecular mechanisms, highlighting the generalization capabilities of sequence-based models in predicting splice sites. These models capture high-level representations of the sequence determinants influencing splice site recognition. However, certain aspects deserve further discussion.

We found relatively lower performance for exonic-like variants that likely affect splicing regulatory elements (Figure 4.5). This may be attributed to the functional role of these elements being context-dependent, a factor not explicitly considered in the tested models. Additionally, other sources of information may have obscured the signal. For instance, the original SpliceAI publication demonstrated that the model implicitly learned the positioning of nucleosomes (fundamental units of chromatin) and used this information as a determinant for exon definition, with regions of higher nucleosome occupancy correlating with higher SpliceAI scores.

For this class of variants, models specifically designed to predict RBP binding, such as those proposed by Avsec *et al.* [268] and Ghanbari and Ohler [269], could be considered. However,

these models are framed as binary classification tasks, predicting whether a given sequence is bound or unbound by a specific RBP, thus losing spatial resolution on the exact binding site. Recently, RBPNet was introduced, predicting experimental CLIP-seq signal at single-nucleotide resolution, thereby overcoming the limitations of classification-based methods [270]. Notably, RBPNet predictions were sensitive to splicing-associated variants near splice sites. Applying RBPNet to our set of deep intronic variants within activated pseudoexons could be a valuable future experiment.

The choice of benchmarking datasets is another important aspect to discuss. We employed a conventional, time-consuming approach by manually curating deep intronic experimentally validated disease-causing variants from the literature (Figure A.2). Additionally, we aggregated and harmonized splicing-altering variants from multiple recent curation efforts, some parallel to ours (Table 4.2). While valuable, this approach cannot scale genome wide and falls short on the generation of negatives. Researchers tend to prioritize, test, and publish variants with observed effects (positives), resulting in a scarcity of negative experimentally-validated data. To address this, an orthogonal source of truth, such as large-scale saturated screens, are emerging alternatives [271]. Massively parallel splicing assays (MPSAa) target all SNVs in and around selected exons, simultaneously phenotyping cells carrying hundreds or thousands of genetic variants of single genes. These assays can even create combinatorial libraries to assess the effect of multiple variants simultaneously [258] - something not considered in our benchmarks. Although these datasets are still scarce for splicing studies [238, 272–274], their availability is expected to increase. As they become more prevalent, they will provide a highly valuable resource for model benchmarking.

8.2 Moving beyond SpliceAI

We addressed model interpretability from various angles. In Chapter 5, we approached model interpretability by focusing on SpliceAI, arguably the most widely used model for splicing-associated analysis, by doing model ablations of RBP motifs. Our findings revealed that SpliceAI indeed uses RBP motifs as predictive features, but caution is necessary when extrapolating these results to broader biological contexts. In particular, additional experiments could probe the specificity of the results, such as ensuring that our RBP-specific knockdown-sensitive sequences are indeed more influenced by the target RBP motifs than by motifs of other RBPs, or even random perturbations.

With the analysis in Chapter 6, we demonstrated that combinations of random perturbations could effectively span the prediction space of the model. It remains unclear whether this prediction coverage is strictly due to sequence motifs or if it involves other layers of splicing regulation implicitly learned by the model, such as RNA structure or splicing kinetics. SpliceAI's lack of explicit training on quantitative cell measurements (e.g., PSI, splicing efficiency)

complicates its use for understanding RBP logic in splicing regulation. When using Pangolin as an oracle - which predicts splice site usage and was trained with RNA-Seq data - we observed more challenges in navigating the prediction space (Figure 6.14). This observation suggests that Pangolin might provide a more realistic landscape of splice site usage *in vivo*.

A recent modular architecture was proposed to enhance splicing prediction with a focus on interpretability by explicitly modeling the binding locations of RBPs through an Adjusted Motif component [275]. This component was trained using position-specific affinity matrices derived from RNA Bind-N-Seq (RBNS) data for 79 RBPs, an in vitro assay that characterizes RBP-RNA interactions. The model enforces sparsity to limit the fraction of sequence positions bound by an RBP, focusing only on the positions with the highest affinity. While this approach aims to provide interpretability by identifying which RBPs contribute to the model's splicing predictions, its overall performance was considerably lower compared to SpliceAI. Moreover, the use of hard constraints that force the model to rely solely on previously known motifs restricts its ability to discover new motifs and uncover novel splicing regulators.

8.3 Grammars express the language of the problem domain

We posit that Genetic Programming constrained by grammars can be a powerful alternative for explainable AI. Our research demonstrates that GGGP can efficiently guide the search for semantically-rich synthetic datasets (Chapter 6). Interestingly, the quality of the search heavily depends on the grammar used, with a less constrained grammar employing random perturbations outperforming a more fine-grained, motif-based grammar (Figure 6.13). Regardless, assuming the model serves as an accurate oracle for the problem, these grammar-guided synthetic datasets can be leveraged to extract valuable biological insights.

The intrinsic interpretability encoded in each population individual's phenotype, defined by the grammar's production rules, enables immediate 'what-if' counterfactuals of model responses at the individual sequence level. This approach bridges the gap between the model and domain experts, contrasting with current attribution methods. While these methods highlight important nucleotides, they require additional analysis to map these low-level features into higher-level concepts like motifs. To facilitate model interrogations, we have developed a flexible software package that abstracts many technical details, empowering users to interact with the model in a domain-appropriate manner (Chapter 7).

A natural next step, which we did not explore due to time constraints, is generating explanations from the synthetic local dataset. This would involve designing a grammar that encodes domain-specific building blocks or concepts, guiding the evolutionary search toward the most discriminative features within the grammar's conceptual framework. For example, the grammar could incorporate production rules representing splicing regulation mechanisms such as motif presence, spacing, or splice site competition. The search would then explore optimal

configurations of these concepts to explain the model's predictions.

Furthermore, we could enhance the grammar's expressiveness by integrating additional constraints that align with human understanding. For instance, specific production rules could cross-reference a motif's location with existing peak data from eCLIP experiments, providing empirical validation for binding site predictions [276]. Similarly, we could flag evolutionarily conserved regions within the sequence [277], suggesting functional importance based on selective pressure - a principle well-established in biology.

In this framework, each individual in the population serves dual roles: as a local surrogate of the oracle (e.g., predicting splice site probability for a cassette exon) and as an intrinsically interpretable entity. The tree-based structure of GP, combined with the domain-aligned concepts encoded in the grammar, renders the model's phenotype an explanation in itself. This approach connects the latent space of the deep learning oracle with the mechanistic reasoning that biologists seek. By presenting explanations in terms of biological constructs, greater trust in these models can be established [278].

8.4 The rise of Large Language Models of DNA

Large Language Models (LLMs) have demonstrated remarkable capabilities in several NLP [2, 279] and protein tasks [280, 281]. Their applications in genomics have followed [282–285], offering a promising new paradigm to decode functionality from DNA/RNA sequences, particularly in non-coding regions of the genome, which are often sparsely covered by short sequence motifs in seemingly random DNA. Unlike traditional sequence-based CNNs, these models are trained through self-supervised learning to encode vast amounts of unlabeled sequences (from human genomes or across different species) into rich contextual embeddings. Such embeddings capture information about each token (which can be a single nucleotide [283] or k-mers of fixed [282] or variable size [285]) and their relationships within the sequence. Consequently, these embeddings can be leveraged for predictive downstream tasks, such as splice site prediction [282, 283].

Base architectures include BERT-based models like DNABERT [286] and Nucleotide Transformer [282], which scale poorly with sequence length, despite self attention alternatives like flash attention applied in DNABERT-2 [285]. Convolutional-based alternatives like GPN [287], HyenaDNA [283], and Evo [288] have been developed, with the latter two utilizing implicit convolutions via Hyena operators [289] to achieve single nucleotide resolution tokenization and efficient scaling of sequence size. The release of these models brings both exciting opportunities and significant challenges, which are discussed next.

As more human genomes representing diverse population ancestries become available, and as more species are sequenced, it is expected that pre-training will incorporate unprecedented diversity. This could lead to the development of genome-wide biologically meaningful

representations that the pre-training burden will handle. These representations can then be utilized for any downstream task, assuming sufficient task-specific data is available. However, a recent study has shown that providing pre-trained embeddings (not fine-tuned) as input for baseline CNNs in several functional genomics tasks provided no advantage over standard one-hot encoded sequences [290]. This suggests that cell-type specific *cis*-regulatory mechanisms needed for a task must be learned during fine-tuning, and the costly pre-training provides no real benefit. While fine-tuning (adjusting the weights of the LLM) is expected to improve performance in downstream tasks, fine-tuning billion-parameter models requires large computational resources that may not be affordable to academic labs. Additionally, interpreting such models becomes even more challenging.

Recent applications of these models to splicing prediction tasks have been explored. For instance, SpliceBERT [291] was pre-trained with RNA sequences of 72 vertebrates and, as the name suggests, its architecture and modeling objectives are based on BERT. The model showed moderate performance on zero-shot variant effect prediction, but benchmarks were not performed against state-of-the-art supervised models like SpliceAI. Other foundational RNA models predicting RNA-Seq coverage [292, 293] have been benchmarked for splicing prediction tasks; however, the lack of single-nucleotide resolution outputs makes it harder to measure variant effects or predict splicing PSI. A new model, SegmentNT [294], builds on top of Nucleotide Transformer and proposes a segmentation model (a 1D U-Net architecture [295]) that predicts DNA sequences up to 30kb long to predict 14 different classes of genomic elements, including splice site and exon probabilities, at single-nucleotide resolution. SegmentNT performed on par with SpliceAI in predicting splice sites, but it extends SpliceAI by also predicting full exon and intron probabilities, allowing for direct assessment of potential transcript isoform switching due to variants directly from the sequence, which we would be very interested to test. However, like SpliceAI, these predictions are cell type agnostic, making it still not possible to study *cis*-regulatory tissue-specific splicing mechanisms.

8.5 Conclusion

We have presented a series of studies exploring the application of sequence-based deep learning models in RNA splicing research. We argue that a model capturing cell type-specific splicing regulation mechanisms is essential for advancing past the prediction of constitutive vs. alternative splicing. With cell-type specific data that extends beyond the heavily used GTEx and ENCODE datasets, the rise of population-scale trained foundational models, additional modalities capturing RBP-RNA interactions, and further maturation of interpretability frameworks (including ours), we envision that in the not-too-distant future, we will have models that truly capture the *cis*-regulatory code of RNA splicing.

Importantly, with the rise of genomics LLMs, establishing more independent benchmarks

becomes critical. It appears that two waves of approaches are competing for state-of-the-art genomics modeling now. On one hand, large companies with substantial resources are training billion-parameter models on vast amounts of data and using this knowledge for practical downstream applications. On the other hand, foundational models trained on shorter sequence regions targeted at specific tasks (e.g., 3'UTRs [296]) or standard supervised CNNs [297, 298] still appear today as essential tools for modeling functional genomics data and interpreting its *cis*-regulatory code. The future holds exciting possibilities, and we look forward to seeing how these research developments will shape biological research in the coming years.

References

[1] A. M. Ozbayoglu, M. U. Gudelek, and O. B. Sezer, *Deep Learning for Financial Applications : A Survey*, Feb. 2020. DOI: 10.48550/arXiv.2002.05786. arXiv: 2002.05786 [cs, q-fin, stat] (cit. on p. 1).

[2] J. Wei *et al.*, *Emergent Abilities of Large Language Models*, Oct. 2022. DOI: 10.48550/arXiv.2206.07682. arXiv: 2206.07682 [cs] (cit. on pp. 1, 118).

[3] A. Esteva *et al.*, "A guide to deep learning in healthcare," *Nat Med*, vol. 25, no. 1, pp. 24–29, Jan. 2019. DOI: 10.1038/s41591-018-0316-z (cit. on p. 1).

[4] A. Esteva *et al.*, "Deep learning-enabled medical computer vision," *npj Digit. Med.*, vol. 4, no. 1, pp. 1–9, Jan. 2021. DOI: 10.1038/s41746-020-00376-2 (cit. on p. 1).

[5] J. Zhou and O. G. Troyanskaya, "Predicting effects of noncoding variants with deep learning–based sequence model," *Nature Methods*, vol. 12, no. 10, pp. 931–934, Oct. 2015. DOI: 10.1038/nmeth.3547 (cit. on pp. 1, 2, 18).

[6] J. Zhou, C. L. Theesfeld, K. Yao, K. M. Chen, A. K. Wong, and O. G. Troyanskaya, "Deep learning sequence-based ab initio prediction of variant effects on expression and disease risk," *Nat Genet*, vol. 50, no. 8, pp. 1171–1179, Aug. 2018. DOI: 10.1038/s41588-018-0160-6 (cit. on pp. 1, 14).

[7] F. E. Baralle and J. Giudice, "Alternative splicing as a regulator of development and tissue identity," *Nat Rev Mol Cell Biol*, vol. 18, no. 7, pp. 437–451, Jul. 2017. DOI: 10.1038/nrm.2017.27 (cit. on pp. 1, 9, 10).

[8] A. Anna and G. Monika, "Splicing mutations in human genetic disorders: Examples, detection, and confirmation," *J Appl Genet*, vol. 59, no. 3, pp. 253–268, 2018. DOI: 10.1007/s13353-018-0444-7 (cit. on pp. 1, 11).

[9] O. Anczuków and A. R. Krainer, "Splicing-factor alterations in cancers," *RNA*, vol. 22, no. 9, pp. 1285–1301, Sep. 2016. DOI: 10.1261/rna.057919.116 (cit. on p. 1).

[10] J. X. Chong *et al.*, "The Genetic Basis of Mendelian Phenotypes: Discoveries, Challenges, and Opportunities," *Am J Hum Genet*, vol. 97, no. 2, pp. 199–215, Aug. 2015. DOI: 10.1016/j.ajhg.2015.06.009 (cit. on p. 1).

[11] S. Srivastava *et al.*, "Meta-analysis and multidisciplinary consensus statement: Exome sequencing is a first-tier clinical diagnostic test for individuals with neurodevelopmental disorders," *Genet Med*, vol. 21, no. 11, pp. 2413–2421, Nov. 2019. DOI: 10.1038/s41436-019-0554-6 (cit. on p. 1).

[12] K. J. Karczewski *et al.*, "The mutational constraint spectrum quantified from variation in 141,456 humans," *Nature*, vol. 581, no. 7809, pp. 434–443, May 2020. DOI: 10.1038/s41586-020-2308-7 (cit. on pp. 1, 41, 44).

[13] D. Taliun *et al.*, "Sequencing of 53,831 diverse genomes from the NHLBI TOPMed Program," *Nature*, vol. 590, no. 7845, pp. 290–299, Feb. 2021. DOI: 10.1038/s41586-021-03205-y (cit. on p. 1).

[14] A. Piovesan, F. Antonaros, L. Vitale, P. Strippoli, M. C. Pelleri, and M. Caracausi, "Human protein-coding genes and gene feature statistics in 2019," *BMC Research Notes*, vol. 12, no. 1, p. 315, Jun. 2019. DOI: 10.1186/s13104-019-4343-8 (cit. on p. 1).

[15] S. Shepard, M. McCreary, and A. Fedorov, "The Peculiarities of Large Intron Splicing in Animals," *PLoS One*, vol. 4, no. 11, e7853, Nov. 2009. DOI: 10.1371/journal.pone.0007853 (cit. on p. 1).

[16] K. Eilbeck, A. Quinlan, and M. Yandell, "Settling the score: Variant prioritization and Mendelian disease," *Nat Rev Genet*, vol. 18, no. 10, pp. 599–612, Oct. 2017. DOI: 10.1038/nrg.2017.52 (cit. on pp. 2, 30).

[17] J. Lord and D. Baralle, "Splicing in the Diagnosis of Rare Disease: Advances and Challenges," *Frontiers in Genetics*, vol. 12, 2021 (cit. on pp. 2, 30).

[18] S. Chakraborty *et al.*, "Interpretability of deep learning models: A survey of results," in *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, Aug. 2017, pp. 1–6. DOI: 10.1109/UIC-ATC.2017.8397411 (cit. on p. 2).

[19] A. Ghorbani, J. Wexler, J. Y. Zou, and B. Kim, "Towards automatic concept-based explanations," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019 (cit. on p. 2).

[20] P. W. Koh *et al.*, "Concept Bottleneck Models," in *Proceedings of the 37th International Conference on Machine Learning*, PMLR, Nov. 2020, pp. 5338–5348 (cit. on p. 2).

[21] G. Novakovsky, O. Fornes, M. Saraswat, S. Mostafavi, and W. W. Wasserman, "ExplaiNN: Interpretable and transparent neural networks for genomics," *Genome Biology*, vol. 24, no. 1, p. 154, Jun. 2023. DOI: 10.1186/s13059-023-02985-y (cit. on pp. 2, 22).

[22] S. Nair, A. Shrikumar, J. Schreiber, and A. Kundaje, "fastISM: Performant in silico saturation mutagenesis for convolutional neural networks," *Bioinformatics*, vol. 38, no. 9, pp. 2397–2403, Apr. 2022. DOI: 10.1093/bioinformatics/btac135 (cit. on pp. 2, 18).

[23] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, PMLR, 2017-08-06/2017-08-11, pp. 3145–3153 (cit. on pp. 2, 78).

[24] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, PMLR, 2017-08-06/2017-08-11, pp. 3319–3328 (cit. on pp. 2, 20).

[25] A. Jha, J. K Aicher, M. R Gazzara, D. Singh, and Y. Barash, "Enhanced Integrated Gradients: Improving interpretability of deep learning models using splicing codes as a case study," *Genome Biology*, vol. 21, no. 1, p. 149, Jun. 2020. DOI: 10.1186/s13059-020-02055-7 (cit. on pp. 2, 20).

[26] Ž. Avsec *et al.*, "Base-resolution models of transcription-factor binding reveal soft motif syntax," *Nature Genetics*, vol. 53, no. 3, pp. 354–366, Mar. 2021. DOI: 10.1038/s41588-021-00782-6 (cit. on pp. 2, 14, 20).

[27] B. P. de Almeida, F. Reiter, M. Pagani, and A. Stark, "DeepSTARR predicts enhancer activity from DNA sequence and enables the de novo design of synthetic enhancers," *Nat Genet*, vol. 54, no. 5, pp. 613–624, May 2022. DOI: 10.1038/s41588-022-01048-5 (cit. on p. 2).

[28] M. Wu, M. C. Hughes, S. Parbhoo, M. Zazzi, V. Roth, and F. Doshi-Velez, *Beyond Sparsity: Tree Regularization of Deep Models for Interpretability*, Nov. 2017. DOI: 10.48550/arXiv.1711.06178 (cit. on p. 2).

[29] B. P. Evans, B. Xue, and M. Zhang, "What's inside the black-box? a genetic programming method for interpreting complex machine learning models," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '19, New York, NY, USA: Association for Computing Machinery, Jul. 2019, pp. 1012–1020. DOI: 10.1145/3321707.3321726 (cit. on pp. 2, 23).

[30] G. Novakovsky, N. Dexter, M. W. Libbrecht, W. W. Wasserman, and S. Mostafavi, "Obtaining genetics insights from deep learning via explainable artificial intelligence," *Nature Reviews Genetics*, pp. 1–13, Oct. 2022. DOI: 10.1038/s41576-022-00532-2 (cit. on pp. 2, 18–22, 53, 78, 82).

[31] M. T. Ribeiro, S. Singh, and C. Guestrin, *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*, Aug. 2016. DOI: 10.48550/arXiv.1602.04938 (cit. on pp. 2, 22, 82).

[32] R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, and F. Giannotti, *Local Rule-Based Explanations of Black Box Decision Systems*, May 2018. DOI: 10.48550/arXiv.1805.10820 (cit. on pp. 2, 23, 82).

[33] L. A. Ferreira, F. G. Guimarães, and R. Silva, *Applying Genetic Programming to Improve Interpretability in Machine Learning Models*, May 2020. DOI: 10.48550/arXiv.2005.09512 (cit. on pp. 2, 82).

[34] E. E. Seitz, D. M. McCandlish, J. B. Kinney, and P. K. Koo, "Interpreting cis-regulatory mechanisms from genomic deep neural networks using surrogate models," *Nat Mach Intell*, vol. 6, no. 6, pp. 701–713, Jun. 2024. DOI: 10.1038/s42256-024-00851-5 (cit. on pp. 2, 22, 82, 83, 90, 95, 175).

[35] L. R. Lopes *et al.*, "Cryptic Splice-Altering Variants in *MYBPC3* Are a Prevalent Cause of Hypertrophic Cardiomyopathy," *Circ: Genomic and Precision Medicine*, vol. 13, no. 3, e002905, Jun. 2020. DOI: 10.1161/CIRCGEN.120.002905 (cit. on pp. 4, 13, 28).

[36] P. Barbosa, M. Ribeiro, M. Carmo-Fonseca, and A. Fonseca, "Clinical significance of genetic variation in hypertrophic cardiomyopathy: Comparison of computational tools to prioritize missense variants," *Frontiers in Cardiovascular Medicine*, vol. 9, 2022 (cit. on pp. 4, 29, 154).

[37] P. Barbosa, R. Savisaar, M. Carmo-Fonseca, and A. Fonseca, "Computational prediction of human deep intronic variation," *GigaScience*, vol. 12, giad085, Oct. 2023. DOI: 10.1093/gigascience/giad085 (cit. on pp. 4, 29).

[38] K. Jaganathan *et al.*, "Predicting Splicing from Primary Sequence with Deep Learning," *Cell*, vol. 176, no. 3, 535–548.e24, Jan. 2019. DOI: 10.1016/j.cell.2018.12.015 (cit. on pp. 4, 10, 25, 26, 30, 34, 70, 89).

[39] G. Espada, L. Ingelse, P. Canelas, P. Barbosa, and A. Fonseca, "Data Types as a More Ergonomic Frontend for Grammar-Guided Genetic Programming," in *Proceedings of the 21st ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences*, ser. GPCE 2022, New York, NY, USA: Association for Computing Machinery, Dec. 2022, pp. 86–94. DOI: 10.1145/3564719.3568697 (cit. on pp. 5, 81, 85, 88, 89, 105).

[40] P. Barbosa, R. Savisaar, and A. Fonseca, "Semantically Rich Local Dataset Generation for Explainable AI in Genomics," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '24, New York, NY, USA: Association for Computing Machinery, Jul. 2024. DOI: 10.1145/3638529.3653990 (cit. on pp. 5, 81).

[41] J. Desterro, P. Bak-Gordon, and M. Carmo-Fonseca, "Targeting mRNA processing as an anticancer strategy," *Nat Rev Drug Discov*, vol. 19, no. 2, pp. 112–129, Feb. 2020. DOI: 10.1038/s41573-019-0042-3 (cit. on pp. 8, 10, 11).

[42] S. Jeong, "SR Proteins: Binders, Regulators, and Connectors of RNA," *Mol Cells*, vol. 40, no. 1, pp. 1–9, Jan. 2017. DOI: 10.14348/molcells.2017.2319 (cit. on pp. 8, 71).

[43] X.-D. Fu and M. Ares, "Context-dependent control of alternative splicing by RNA-binding proteins," *Nat Rev Genet*, vol. 15, no. 10, pp. 689–701, Oct. 2014. DOI: 10.1038/nrg3778 (cit. on pp. 8, 10, 62, 71, 74, 78).

[44] M. C. Wahl, C. L. Will, and R. Lührmann, "The Spliceosome: Design Principles of a Dynamic RNP Machine," *Cell*, vol. 136, no. 4, pp. 701–718, Feb. 2009. DOI: 10.1016/j.cell.2009.02.009 (cit. on p. 7).

[45] J. Chen and W. A. Weiss, "Alternative splicing in cancer: Implications for biology and therapy," *Oncogene*, vol. 34, no. 1, pp. 1–14, Jan. 2015. DOI: 10.1038/onc.2013.570 (cit. on p. 9).

[46] J. Ule and B. J. Blencowe, "Alternative Splicing Regulatory Networks: Functions, Mechanisms, and Evolution," *Molecular Cell*, vol. 76, no. 2, pp. 329–345, Oct. 2019. DOI: 10.1016/j.molcel.2019.09.017 (cit. on pp. 10, 11, 79).

[47] S. Erkelenz *et al.*, "Position-dependent splicing activation and repression by SR and hnRNP proteins rely on common mechanisms," *RNA*, vol. 19, no. 1, pp. 96–102, Jan. 2013. DOI: 10.1261/rna.037044.112 (cit. on pp. 10, 71).

[48] M. B. Warf and J. A. Berglund, "The role of RNA structure in regulating pre-mRNA splicing," *Trends Biochem Sci*, vol. 35, no. 3, pp. 169–178, Mar. 2010. DOI: 10.1016/j.tibs.2009.10.004 (cit. on p. 10).

[49] T. W. Nilsen and B. R. Graveley, "Expansion of the eukaryotic proteome by alternative splicing," *Nature*, vol. 463, no. 7280, pp. 457–463, Jan. 2010. DOI: 10.1038/nature08909 (cit. on p. 10).

[50] C. Iannone and J. Valcárcel, "Chromatin's thread to alternative splicing regulation," *Chromosoma*, vol. 122, no. 6, pp. 465–474, Dec. 2013. DOI: 10.1007/s00412-013-0425-x (cit. on p. 10).

[51] B. S. Zhao, I. A. Roundtree, and C. He, "Post-transcriptional gene regulation by mRNA modifications," *Nat Rev Mol Cell Biol*, vol. 18, no. 1, pp. 31–42, Jan. 2017. DOI: 10.1038/nrm.2016.132 (cit. on p. 10).

[52] G.-S. Wang and T. A. Cooper, "Splicing in disease: Disruption of the splicing code and the decoding machinery," *Nat Rev Genet*, vol. 8, no. 10, pp. 749–761, Oct. 2007. DOI: 10.1038/nrg2164 (cit. on p. 10).

[53] K. H. Lim, L. Ferraris, M. E. Filloux, B. J. Raphael, and W. G. Fairbrother, "Using positional distribution to identify splicing elements and predict pre-mRNA processing defects in human genes," *Proceedings of the National Academy of Sciences*, vol. 108, no. 27, pp. 11 093–11 098, Jul. 2011. DOI: 10.1073/pnas.1101135108 (cit. on p. 10).

[54] M. J. Landrum *et al.*, "ClinVar: Improving access to variant interpretations and supporting evidence," *Nucleic Acids Research*, vol. 46, no. D1, pp. D1062–D1067, Jan. 2018. DOI: 10.1093/nar/gkx1153 (cit. on p. 10).

[55] P. D. Stenson *et al.*, "The Human Gene Mutation Database (HGMD®): Optimizing its use in a clinical diagnostic or research setting," *Hum Genet*, vol. 139, no. 10, pp. 1197–1207, 2020. DOI: 10.1007/s00439-020-02199-3 (cit. on p. 10).

[56] J. Lord *et al.*, "Pathogenicity and selective constraint on variation near splice sites," *Genome Res.*, vol. 29, no. 2, pp. 159–170, Feb. 2019. DOI: 10.1101/gr.238444.118 (cit. on p. 11).

[57] A. J. M. Blakes *et al.*, "A systematic analysis of splicing variants identifies new diagnoses in the 100,000 Genomes Project," *Genome Medicine*, vol. 14, no. 1, p. 79, Jul. 2022. DOI: 10.1186/s13073-022-01087-x (cit. on p. 11).

[58] C. R. Sibley, L. Blazquez, and J. Ule, "Lessons from non-canonical splicing," *Nat Rev Genet*, vol. 17, no. 7, pp. 407–421, Jul. 2016. DOI: 10.1038/nrg.2016.46 (cit. on pp. 11, 153).

[59] R. Vaz-Drago, N. Custódio, and M. Carmo-Fonseca, "Deep intronic mutations and human disease," *Hum Genet*, vol. 136, no. 9, pp. 1093–1111, Sep. 2017. DOI: 10.1007/s00439-017-1809-4 (cit. on pp. 11, 41, 42, 44, 152, 155, 159).

[60] N. P. Keegan, S. D. Wilton, and S. Fletcher, "Analysis of Pathogenic Pseudoexons Reveals Novel Mechanisms Driving Cryptic Splicing," *Frontiers in Genetics*, vol. 12, 2022 (cit. on pp. 11, 34, 44).

[61] J. M. Ellingford *et al.*, "Recommendations for clinical interpretation of variants found in non-coding regions of the genome," *Genome Med*, vol. 14, no. 1, p. 73, Jul. 2022. DOI: 10.1186/s13073-022-01073-3 (cit. on p. 12).

[62] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. DOI: 10.1038/nature14539 (cit. on p. 13).

[63] M. Abadi *et al.*, *TensorFlow: A system for large-scale machine learning*, May 2016. DOI: 10.48550/arXiv.1605.08695. arXiv: 1605.08695 [cs] (cit. on p. 13).

[64] A. Paszke *et al.*, *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, Dec. 2019. DOI: 10.48550/arXiv.1912.01703. arXiv: 1912.01703 [cs, stat] (cit. on p. 13).

[65] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, vol. 25, Curran Associates, Inc., 2012 (cit. on p. 14).

[66] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep Learning for Computer Vision: A Brief Review," *Computational Intelligence and Neuroscience*, vol. 2018, p. 7 068 349, Feb. 2018. DOI: 10.1155/2018/7068349 (cit. on p. 14).

[67] A. Vaswani *et al.*, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017 (cit. on p. 14).

[68] S. Islam *et al.*, "A comprehensive survey on applications of transformers for deep learning tasks," *Expert Systems with Applications*, vol. 241, p. 122 666, May 2024. DOI: 10.1016/j.eswa.2023.122666 (cit. on p. 14).

[69] H. Wang *et al.*, "Scientific discovery in the age of artificial intelligence," *Nature*, vol. 620, no. 7972, pp. 47–60, Aug. 2023. DOI: 10.1038/s41586-023-06221-2 (cit. on p. 14).

[70] S. E. Whang, Y. Roh, H. Song, and J.-G. Lee, "Data collection and quality challenges in deep learning: A data-centric AI perspective," *The VLDB Journal*, vol. 32, no. 4, pp. 791–813, Jul. 2023. DOI: 10.1007/s00778-022-00775-9 (cit. on p. 14).

[71] M. Pandey *et al.*, "The transformational role of GPU computing and deep learning in drug discovery," *Nature Machine Intelligence*, vol. 4, no. 3, pp. 211–221, Mar. 2022. DOI: 10.1038/s42256-022-00463-x (cit. on p. 14).

[72] M. Krenn *et al.*, "On scientific understanding with artificial intelligence," *Nature Reviews Physics*, vol. 4, no. 12, pp. 761–769, Dec. 2022. DOI: 10.1038/s42254-022-00518-3 (cit. on p. 14).

[73] A. Maslova *et al.*, "Deep learning of immune cell differentiation," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 117, no. 41, pp. 25 655–25 666, Oct. 2020. DOI: 10.1073/pnas.2011795117 (cit. on p. 14).

[74] S. Ruder, *An overview of gradient descent optimization algorithms*, Jun. 2017. arXiv: 1609.04747 [cs] (cit. on p. 15).

[75] D. E. Rumelhart, J. L. McClelland, and P. R. Group, *Parallel Distributed Processing, Volume 1: Explorations in the Microstructure of Cognition: Foundations*. The MIT Press, Jul. 1986. DOI: 10.7551/mitpress/5236.001.0001 (cit. on p. 15).

[76] G. Eraslan, Ž. Avsec, J. Gagneur, and F. J. Theis, "Deep learning: New computational modelling techniques for genomics," *Nature Reviews Genetics*, vol. 20, no. 7, pp. 389–403, Jul. 2019. DOI: 10.1038/s41576-019-0122-6 (cit. on pp. 17, 30).

[77] F. Yu and V. Koltun, *Multi-Scale Context Aggregation by Dilated Convolutions*, Apr. 2016. arXiv: 1511.07122 [cs] (cit. on p. 16).

[78] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014 (cit. on p. 17).

[79] S. Ioffe and C. Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, Mar. 2015. DOI: 10.48550/arXiv.1502.03167. arXiv: 1502.03167 [cs] (cit. on p. 17).

[80] D. R. Kelley, J. Snoek, and J. L. Rinn, "Basset: Learning the regulatory code of the accessible genome with deep convolutional neural networks," *Genome Research*, vol. 26, no. 7, pp. 990–999, Jul. 2016. DOI: 10.1101/gr.200535.115 (cit. on pp. 18, 20).

[81] P. K. Koo and M. Ploenzke, "Improving representations of genomic sequence motifs in convolutional networks with exponential activations," *Nature machine intelligence*, vol. 3, no. 3, pp. 258–266, Mar. 2021. DOI: 10.1038/s42256-020-00291-x (cit. on p. 18).

[82] Ž. Avsec *et al.*, "The Kipoi repository accelerates community exchange and reuse of predictive models for genomics," *Nature Biotechnology*, vol. 37, no. 6, pp. 592–600, 2019. DOI: 10.1038/s41587-019-0140-0 (cit. on pp. 19, 31, 34, 55, 154).

[83] C. B. Azodi, J. Tang, and S.-H. Shiu, "Opening the Black Box: Interpretable Machine Learning for Geneticists," *Trends in Genetics*, vol. 36, no. 6, pp. 442–455, Jun. 2020. DOI: 10.1016/j.tig.2020.03.005 (cit. on p. 19).

[84] K. Simonyan, A. Vedaldi, and A. Zisserman, *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*, Apr. 2014. DOI: 10.48550/arXiv.1312.6034. arXiv: 1312.6034 [cs] (cit. on p. 19).

[85] A. Shrikumar, P. Greenside, A. Shcherbina, and A. Kundaje, *Not Just a Black Box: Learning Important Features Through Propagating Activation Differences*, Apr. 2017. arXiv: 1605.01713 [cs] (cit. on p. 19).

[86] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Cham: Springer International Publishing, 2014, pp. 818–833. DOI: 10.1007/978-3-319-10590-1_53 (cit. on p. 19).

[87] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, *Striving for Simplicity: The All Convolutional Net*, Apr. 2015. DOI: 10.48550/arXiv.1412.6806. arXiv: 1412.6806 [cs] (cit. on p. 19).

[88] A. Shrikumar, P. Greenside, and A. Kundaje, *Learning Important Features Through Propagating Activation Differences*, Oct. 2019. arXiv: 1704.02685 [cs] (cit. on p. 20).

[89] S. Lundberg and S.-I. Lee, *A Unified Approach to Interpreting Model Predictions*, Nov. 2017. DOI: 10.48550/arXiv.1705.07874. arXiv: 1705.07874 [cs, stat] (cit. on p. 20).

[90] A. Shrikumar *et al.*, *Technical Note on Transcription Factor Motif Discovery from Importance Scores (TF-MoDISco) version 0.5.6.5*, Apr. 2020. DOI: 10.48550/arXiv.1811.00416. arXiv: 1811.00416 [cs, q-bio, stat] (cit. on p. 20).

[91] P. Greenside, T. Shimko, P. Fordyce, and A. Kundaje, "Discovering epistatic feature interactions from neural network models of regulatory DNA sequences," *Bioinformatics*, vol. 34, no. 17, pp. i629–i637, Sep. 2018. DOI: 10.1093/bioinformatics/bty575 (cit. on p. 21).

[92] S. Toneyan and P. K. Koo, "Interpreting cis-regulatory interactions from large-scale deep neural networks," *Nature Genetics*, pp. 1–11, Sep. 2024. DOI: 10.1038/s41588-024-01923-3 (cit. on p. 21).

[93] J. Ma *et al.*, "Using deep learning to model the hierarchical structure and function of a cell," *Nature Methods*, vol. 15, no. 4, pp. 290–298, Apr. 2018. DOI: 10.1038/nmeth.4627 (cit. on p. 21).

[94] H. A. Elmarakeby *et al.*, "Biologically informed deep neural network for prostate cancer discovery," *Nature*, vol. 598, no. 7880, pp. 348–352, Oct. 2021. DOI: 10.1038/s41586-021-03922-4 (cit. on p. 21).

[95] D. Quang and X. Xie, "DanQ: A hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences," *Nucleic Acids Research*, vol. 44, no. 11, e107, Jun. 2016. DOI: 10.1093/nar/gkw226 (cit. on p. 21).

[96] A. T. Balcı, M. M. Ebeid, P. V. Benos, D. Kostka, and M. Chikina, "An intrinsically interpretable neural network architecture for sequence-to-function learning," *Bioinformatics*, vol. 39, no. Supplement_1, pp. i413–i422, Jun. 2023. DOI: 10.1093/bioinformatics/btad271 (cit. on p. 21).

[97] S. E. Liao, M. Sudarshan, and O. Regev, "Deciphering RNA splicing logic with interpretable machine learning," *Proceedings of the National Academy of Sciences*, vol. 120, no. 41, e2221165120, Oct. 2023. DOI: 10.1073/pnas.2221165120 (cit. on p. 21).

[98] R. Agarwal *et al.*, *Neural Additive Models: Interpretable Machine Learning with Neural Nets*, Oct. 2021. arXiv: 2004.13912 [cs, stat] (cit. on p. 22).

[99] C. Molnar, *Interpretable Machine Learning*. 2019 (cit. on p. 22).

[100] D. Alvarez-Melis and T. S. Jaakkola, *On the Robustness of Interpretability Methods*, Jun. 2018. arXiv: 1806.08049 [cs, stat] (cit. on p. 22).

[101] A. Tareen, M. Kooshkbaghi, A. Posfai, W. T. Ireland, D. M. McCandlish, and J. B. Kinney, "MAVE-NN: Learning genotype-phenotype maps from multiplex assays of variant effect," *Genome Biology*, vol. 23, no. 1, p. 98, Apr. 2022. DOI: 10.1186/s13059-022-02661-7 (cit. on p. 22).

[102] B. Wang, W. Pei, B. Xue, and M. Zhang, "Evolving local interpretable model-agnostic explanations for deep neural networks in image classification," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '21, New York, NY, USA: Association for Computing Machinery, Jul. 2021, pp. 173–174. DOI: 10.1145/3449726.3459452 (cit. on p. 22).

[103] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992 (cit. on pp. 23, 83).

[104] P. A. Whigham, "Grammatically-based genetic programming," in *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications*, J. P. Rosca, Ed., Tahoe City, California, USA, Sep. 1995, pp. 33–41 (cit. on pp. 23, 85).

[105] D. Deshpande *et al.*, "RNA-seq data science: From raw data to effective interpretation," *Frontiers in Genetics*, vol. 14, p. 997 383, Mar. 2023. DOI: 10.3389/fgene.2023.997383 (cit. on p. 24).

[106] T. Steijger *et al.*, "Assessment of transcript reconstruction methods for RNA-seq," *Nature Methods*, vol. 10, no. 12, pp. 1177–1184, Dec. 2013. DOI: 10.1038/nmeth.2714 (cit. on p. 24).

[107] A. Roberts, C. Trapnell, J. Donaghey, J. L. Rinn, and L. Pachter, "Improving RNA-Seq expression estimates by correcting for fragment bias," *Genome Biology*, vol. 12, no. 3, R22, Mar. 2011. DOI: 10.1186/gb-2011-12-3-r22 (cit. on p. 24).

[108] S. Shen *et al.*, "rMATS: Robust and flexible detection of differential alternative splicing from replicate RNA-Seq data," *Proceedings of the National Academy of Sciences*, vol. 111, no. 51, E5593–E5601, Dec. 2014. DOI: 10.1073/pnas.1419161111 (cit. on pp. 24, 58).

[109] J. Vaquero-Garcia *et al.*, "A new view of transcriptome complexity and regulation through the lens of local splicing variations," *eLife*, vol. 5, J. Valcárcel, Ed., e11752, Feb. 2016. DOI: 10.7554/eLife.11752 (cit. on p. 24).

[110] M. Ascensão-Ferreira, R. Martins-Silva, N. Saraiva-Agostinho, and N. L. Barbosa-Morais, "betAS: Intuitive analysis and visualization of differential alternative splicing using beta distributions," *RNA (New York, N.Y.)*, vol. 30, no. 4, pp. 337–353, Mar. 2024. DOI: 10.1261/rna.079764.123 (cit. on p. 24).

[111] J. Ule, K. Jensen, A. Mele, and R. B. Darnell, "CLIP: A method for identifying protein-RNA interaction sites in living cells," *Methods (San Diego, Calif.)*, vol. 37, no. 4, pp. 376–386, Dec. 2005. DOI: 10.1016/j.ymeth.2005.07.018 (cit. on p. 24).

[112] E. L. Van Nostrand *et al.*, "Robust transcriptome-wide discovery of RNA-binding protein binding sites with enhanced CLIP (eCLIP)," *Nature Methods*, vol. 13, no. 6, pp. 508–514, Jun. 2016. DOI: 10.1038/nmeth.3810 (cit. on p. 24).

[113] E. L. Van Nostrand *et al.*, "A large-scale binding and functional map of human RNA-binding proteins," *Nature*, vol. 583, no. 7818, pp. 711–719, Jul. 2020. DOI: 10.1038/s41586-020-2077-3 (cit. on pp. 24, 57, 58, 71, 75, 95, 97).

[114] G. Yeo and C. B. Burge, "Maximum entropy modeling of short sequence motifs with applications to RNA splicing signals," *Journal of Computational Biology*, vol. 11, no. 2-3, pp. 377–394, Mar. 2004. DOI: 10.1089/1066527041410418 (cit. on pp. 25, 31, 33).

[115] F.-O. Desmet, D. Hamroun, M. Lalande, G. Collod-Béroud, M. Claustres, and C. Béroud, "Human Splicing Finder: An online bioinformatics tool to predict splicing signals," *Nucleic Acids Research*, vol. 37, no. 9, e67, May 2009. DOI: 10.1093/nar/gkp215 (cit. on p. 25).

[116] M. M. Yin and J. T. L. Wang, "Effective hidden Markov models for detecting splicing junction sites in DNA sequences," *Information Sciences*, Bioinformatics, vol. 139, no. 1, pp. 139–163, Nov. 2001. DOI: 10.1016/S0020-0255(01)00160-8 (cit. on p. 25).

[117] E. Pashaei, M. Ozen, and N. Aydin, "Splice site identification in human genome using random forest," *Health and Technology*, vol. 7, no. 1, pp. 141–152, Mar. 2017. DOI: 10.1007/s12553-016-0157-z (cit. on p. 25).

[118] S. Ke *et al.*, "Quantitative evaluation of all hexamers as exonic splicing elements," *Genome Res.*, vol. 21, no. 8, pp. 1360–1374, Aug. 2011. DOI: 10.1101/gr.119628.110 (cit. on pp. 25, 32, 35, 46).

[119] A. B. Rosenberg, R. P. Patwardhan, J. Shendure, and G. Seelig, "Learning the sequence determinants of alternative splicing from millions of random sequences," *Cell*, vol. 163, no. 3, pp. 698–711, Oct. 2015. DOI: 10.1016/j.cell.2015.09.054 (cit. on pp. 25, 27, 31, 33, 75).

[120] R. Wang, Z. Wang, J. Wang, and S. Li, "SpliceFinder: Ab initio prediction of splice sites using convolutional neural network," *BMC bioinformatics*, vol. 20, no. Suppl 23, p. 652, Dec. 2019. DOI: 10.1186/s12859-019-3306-3 (cit. on p. 25).

[121] Y. Zhang, X. Liu, J. MacLeod, and J. Liu, "Discerning novel splice junctions derived from RNA-seq alignment: A deep learning approach," *BMC Genomics*, vol. 19, no. 1, p. 971, Dec. 2018. DOI: 10.1186/s12864-018-5350-1 (cit. on p. 25).

[122] T. Naito, "Predicting the impact of single nucleotide variants on splicing via sequence-based deep neural networks and genomic features," *Human Mutation*, humu.23794, May 2019. DOI: 10.1002/humu.23794 (cit. on pp. 25, 32, 35).

[123] J. Zuallaert, F. Godin, M. Kim, A. Soete, Y. Saeys, and W. De Neve, "SpliceRover: Interpretable convolutional neural networks for improved splice site prediction," *Bioinformatics (Oxford, England)*, vol. 34, no. 24, pp. 4180–4188, Dec. 2018. DOI: 10.1093/bioinformatics/bty497 (cit. on pp. 25, 32, 35, 46).

[124] N. Scalzitti *et al.*, "Spliceator: Multi-species splice site prediction using convolutional neural networks," *BMC Bioinformatics*, vol. 22, no. 1, p. 561, Nov. 2021. DOI: 10.1186/s12859-021-04471-3 (cit. on pp. 25, 32, 35, 46).

[125] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90 (cit. on p. 25).

[126] A. Frankish *et al.*, "GENCODE reference annotation for the human and mouse genomes," *Nucleic Acids Research*, vol. 47, no. D1, pp. D766–D773, Jan. 2019. DOI: 10.1093/nar/gky955 (cit. on pp. 25, 30).

[127] J. Lonsdale *et al.*, "The Genotype-Tissue Expression (GTEx) project," *Nat Genet*, vol. 45, no. 6, pp. 580–585, Jun. 2013. DOI: 10.1038/ng.2653 (cit. on pp. 25, 34, 45).

[128] K.-H. Chao, A. Mao, S. L. Salzberg, and M. Pertea, "Splam: A deep-learning-based splice site predictor that improves spliced alignments," *Genome Biology*, vol. 25, no. 1, p. 243, Sep. 2024. DOI: 10.1186/s13059-024-03379-4 (cit. on p. 25).

[129] A. Corvelo, M. Hallegger, C. W. J. Smith, and E. Eyras, "Genome-Wide Association between Branch Point Properties and Alternative Splicing," *PLOS Computational Biology*, vol. 6, no. 11, e1001016, Nov. 2010. DOI: 10.1371/journal.pcbi.1001016 (cit. on pp. 26, 32, 35, 44).

[130] Q. Zhang, X. Fan, Y. Wang, M.-a. Sun, J. Shao, and D. Guo, "BPP: A sequence-based algorithm for branch point prediction," *Bioinformatics*, vol. 33, no. 20, pp. 3166–3172, Oct. 2017. DOI: 10.1093/bioinformatics/btx401 (cit. on pp. 26, 32, 35, 44).

[131] P. Zhang *et al.*, "Genome-wide detection of human variants that disrupt intronic branchpoints," *Proceedings of the National Academy of Sciences*, vol. 119, no. 44, e2211194119, Nov. 2022. DOI: 10.1073/pnas.2211194119 (cit. on pp. 26, 35, 43, 44, 152).

[132] J. M. Paggi and G. Bejerano, "A sequence-based, deep learning model accurately predicts RNA splicing branchpoints," *RNA*, vol. 24, no. 12, pp. 1647–1658, Dec. 2018. DOI: 10.1261/rna.066290.118 (cit. on pp. 26, 35, 44).

[133] H. Bretschneider, S. Gandhi, A. G. Deshwar, K. Zuberi, and B. J. Frey, "COSSMO: Predicting competitive alternative splice site selection using deep learning," *Bioinformatics*, vol. 34, no. 13, pp. i429–i437, Jul. 2018. DOI: 10.1093/bioinformatics/bty244 (cit. on p. 27).

[134] J. Cheng *et al.*, "MMSplice: Modular modeling improves the predictions of genetic variant effects on splicing," *Genome Biology*, vol. 20, no. 1, p. 48, Mar. 2019. DOI: 10.1186/s13059-019-1653-z (cit. on pp. 27, 31, 34).

[135] Y. Barash *et al.*, "Deciphering the splicing code," *Nature*, vol. 465, no. 7294, pp. 53–59, May 2010. DOI: 10.1038/nature09000 (cit. on p. 27).

[136] H. Y. Xiong, Y. Barash, and B. J. Frey, "Bayesian prediction of tissue-regulated splicing using RNA sequence and cellular context," *Bioinformatics*, vol. 27, no. 18, pp. 2554–2562, Sep. 2011. DOI: 10.1093/bioinformatics/btr444 (cit. on p. 27).

[137] M. K. K. Leung, H. Y. Xiong, L. J. Lee, and B. J. Frey, "Deep learning of the tissue-regulated splicing code," *Bioinformatics*, vol. 30, no. 12, pp. i121–i129, Jun. 2014. DOI: 10.1093/bioinformatics/btu277 (cit. on p. 27).

[138] H. Y. Xiong *et al.*, "The human splicing code reveals new insights into the genetic determinants of disease," *Science (New York, N.Y.)*, vol. 347, no. 6218, 2015. DOI: 10.1126/science.1254806 (cit. on pp. 27, 31, 33).

[139] A. Jha, M. R. Gazzara, and Y. Barash, "Integrative deep models for alternative splicing," *Bioinformatics*, vol. 33, no. 14, pp. i274–i282, Jul. 2017. DOI: 10.1093/bioinformatics/btx268 (cit. on p. 27).

[140] J. Cheng, M. H. Çelik, A. Kundaje, and J. Gagneur, "MTSplice predicts effects of genetic variants on tissue-specific splicing," *Genome Biology*, vol. 22, no. 1, p. 94, Mar. 2021. DOI: 10.1186/s13059-021-02273-7 (cit. on p. 28).

[141] T. Zeng and Y. I. Li, "Predicting RNA splicing from DNA sequence using Pangolin," *Genome Biology*, vol. 23, no. 1, p. 103, Apr. 2022. DOI: 10.1186/s13059-022-02664-4 (cit. on pp. 28, 30, 34, 102).

[142] N. Wagner *et al.*, "Aberrant splicing prediction across human tissues," *Nature Genetics*, vol. 55, no. 5, pp. 861–870, May 2023. DOI: 10.1038/s41588-023-01373-3 (cit. on pp. 28, 30, 34, 37, 53).

[143] D. N. Cooper, "Functional intronic polymorphisms: Buried treasure awaiting discovery within our genes," *Hum Genomics*, vol. 4, no. 5, pp. 284–288, Jun. 2010. DOI: 10.1186/1479-7364-4-5-284 (cit. on p. 30).

[144] M. Lek *et al.*, "Analysis of protein-coding genetic variation in 60,706 humans," *Nature*, vol. 536, no. 7616, pp. 285–291, Aug. 2016. DOI: 10.1038/nature19057 (cit. on p. 30).

[145] I. Dunham *et al.*, "An integrated encyclopedia of DNA elements in the human genome," *Nature*, vol. 489, no. 7414, pp. 57–74, Sep. 2012. DOI: 10.1038/nature11247 (cit. on p. 30).

[146] M. J. Cormier, B. S. Pedersen, P. Bayrak-Toydemir, and A. R. Quinlan, "Combining genetic constraint with predictions of alternative splicing to prioritize deleterious splicing in rare disease studies," *BMC Bioinformatics*, vol. 23, no. 1, p. 482, Nov. 2022. DOI: 10.1186/s12859-022-05041-x (cit. on pp. 30, 34, 37).

[147] R. Kurosawa *et al.*, "PDIVAS: Pathogenicity predictor for Deep-Intronic Variants causing Aberrant Splicing," *BMC Genomics*, vol. 24, no. 1, p. 601, Oct. 2023. DOI: 10.1186/s12864-023-09645-2 (cit. on pp. 30, 34, 37).

[148] Y. Strauch, J. Lord, M. Niranjan, and D. Baralle, "CI-SpliceAI—Improving machine learning predictions of disease causing splicing variants using curated alternative splice sites," *PLOS ONE*, vol. 17, no. 6, e0269159, Jun. 2022. DOI: 10.1371/journal.pone.0269159 (cit. on pp. 30, 34).

[149] L. M. Weber *et al.*, "Essential guidelines for computational method benchmarking," *Genome Biology*, vol. 20, no. 1, p. 125, Jun. 2019. DOI: 10.1186/s13059-019-1738-8 (cit. on p. 30).

[150] S. Buchka, A. Hapfelmeier, P. P. Gardner, R. Wilson, and A.-L. Boulesteix, "On the optimistic performance evaluation of newly introduced bioinformatic methods," *Genome Biology*, vol. 22, no. 1, p. 152, May 2021. DOI: 10.1186/s13059-021-02365-4 (cit. on p. 30).

[151] R. Leman *et al.*, "Assessment of branch point prediction tools to predict physiological branch points and their alteration by variants," *BMC Genomics*, vol. 21, no. 1, p. 86, Jan. 2020. DOI: 10.1186/s12864-020-6484-5 (cit. on pp. 30, 35, 43, 44, 152).

[152] H. Tubeuf *et al.*, "Large-scale comparative evaluation of user-friendly tools for predicting variant-induced alterations of splicing regulatory elements," *Human Mutation*, vol. 41, no. 10, pp. 1811–1829, 2020. DOI: 10.1002/humu.24091 (cit. on pp. 30, 35, 44).

[153] A. Moles-Fernández, J. Domènech-Vivó, A. Tenés, J. Balmaña, O. Diez, and S. Gutiérrez-Enríquez, "Role of Splicing Regulatory Elements and In Silico Tools Usage in the Identification of Deep Intronic Splicing Variants in Hereditary Breast/Ovarian Cancer Genes," *Cancers (Basel)*, vol. 13, no. 13, p. 3341, Jul. 2021. DOI: 10.3390/cancers13133341 (cit. on pp. 30, 44, 45, 53).

[154] T. V. Riepe, M. Khan, S. Roosing, F. P. M. Cremers, and P. A. C. 't Hoen, "Benchmarking deep learning splice prediction tools using functional splice assays," *Human Mutation*, vol. 42, no. 7, pp. 799–810, 2021. DOI: 10.1002/humu.24212 (cit. on p. 30).

[155] C. Rowlands *et al.*, "Comparison of in silico strategies to prioritize rare genomic variants impacting RNA splicing for the diagnosis of genomic disorders," *Sci Rep*, vol. 11, no. 1, p. 20 607, Oct. 2021. DOI: 10.1038/s41598-021-99747-2 (cit. on pp. 30, 89).

[156] C. Ha, J.-W. Kim, and J.-H. Jang, "Performance Evaluation of SpliceAI for the Prediction of Splicing of NF1 Variants," *Genes*, vol. 12, no. 9, p. 1308, Sep. 2021. DOI: 10.3390/genes12091308 (cit. on pp. 30, 89).

[157] K. Li *et al.*, "Performance evaluation of differential splicing analysis methods and splicing analytics platform construction," *Nucleic Acids Research*, gkac686, Aug. 2022. DOI: 10.1093/nar/gkac686 (cit. on p. 30).

[158] R. Leman *et al.*, "SPiP: Splicing Prediction Pipeline, a machine learning tool for massive detection of exonic and intronic variant effects on mRNA splicing," *Human Mutation*, vol. 43, no. 12, pp. 2308–2323, 2022. DOI: 10.1002/humu.24491 (cit. on pp. 30, 34).

[159] S. Li *et al.*, "CAPICE: A computational method for Consequence-Agnostic Pathogenicity Interpretation of Clinical Exome variations," *Genome Medicine*, vol. 12, no. 1, p. 75, Aug. 2020. DOI: 10.1186/s13073-020-00775-w (cit. on pp. 31, 33).

[160] X. Jian, E. Boerwinkle, and X. Liu, "In silico prediction of splice-altering single nucleotide variants in the human genome," *Nucleic Acids Res*, vol. 42, no. 22, pp. 13 534–13 544, Dec. 2014. DOI: 10.1093/nar/gku1206 (cit. on pp. 31, 33, 34).

[161] H. Liu *et al.*, "Performance evaluation of computational methods for splice-disrupting variants and improving the performance using the machine learning-based framework," *Briefings in Bioinformatics*, vol. 23, no. 5, bbac334, Aug. 2022. DOI: 10.1093/bib/bbac334 (cit. on pp. 31, 34, 37).

[162] K. A. Jagadeesh *et al.*, "S-CAP extends pathogenicity prediction to genetic variants that affect RNA splicing.," *Nature genetics*, vol. 51, no. 4, pp. 755–763, Feb. 2019. DOI: 10.1038/s41588-019-0348-4 (cit. on pp. 32–34).

[163] L. Cartegni, J. Wang, Z. Zhu, M. Q. Zhang, and A. R. Krainer, "ESEfinder: A web resource to identify exonic splicing enhancers," *Nucleic Acids Res*, vol. 31, no. 13, pp. 3568–3571, Jul. 2003. DOI: 10.1093/nar/gkg616 (cit. on pp. 32, 34, 46).

[164] S. Erkelenz, S. Theiss, M. Otte, M. Widera, J. O. Peter, and H. Schaal, "Genomic HEXploring allows landscaping of novel potential splicing regulatory elements," *Nucleic Acids Res*, vol. 42, no. 16, pp. 10 681–10 697, 2014. DOI: 10.1093/nar/gku736 (cit. on pp. 32, 35, 46).

[165] B. J. Livesey *et al.*, *Guidelines for releasing a variant effect predictor*, Apr. 2024. DOI: 10.48550/arXiv.2404.10807. arXiv: 2404.10807 [q-bio] (cit. on p. 32).

[166] A. Siepel *et al.*, "Evolutionarily conserved elements in vertebrate, insect, worm, and yeast genomes," *Genome Res.*, vol. 15, no. 8, pp. 1034–1050, Aug. 2005. DOI: 10.1101/gr.3715005 (cit. on pp. 33, 39).

[167] J. Li *et al.*, "Performance evaluation of pathogenicity-computation methods for missense variants," *Nucleic Acids Res*, vol. 46, no. 15, pp. 7793–7804, Sep. 2018. DOI: 10.1093/nar/gky678 (cit. on p. 33).

[168] A. Siepel, K. S. Pollard, and D. Haussler, "New Methods for Detecting Lineage-Specific Selection," in *Research in Computational Molecular Biology*, A. Apostolico, C. Guerra, S. Istrail, P. A. Pevzner, and M. Waterman, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2006, pp. 190–205. DOI: 10.1007/11732990_17 (cit. on p. 33).

[169] C. Dong *et al.*, "Comparison and integration of deleteriousness prediction methods for nonsynonymous SNVs in whole exome sequencing studies," *Hum Mol Genet*, vol. 24, no. 8, pp. 2125–2137, Apr. 2015. DOI: 10.1093/hmg/ddu733 (cit. on p. 33).

[170] M. Garber, M. Guttman, M. Clamp, M. C. Zody, N. Friedman, and X. Xie, "Identifying novel constrained elements by exploiting biased substitution patterns," *Bioinformatics*, vol. 25, no. 12, pp. i54–i62, Jun. 2009. DOI: 10.1093/bioinformatics/btp190 (cit. on p. 33).

[171] E. V. Davydov, D. L. Goode, M. Sirota, G. M. Cooper, A. Sidow, and S. Batzoglou, "Identifying a High Fraction of the Human Genome to be under Selective Constraint Using GERP++," *PLoS Comput Biol*, vol. 6, no. 12, e1001025, Dec. 2010. DOI: 10.1371/journal.pcbi.1001025 (cit. on p. 33).

[172] H. A. Shihab *et al.*, "An integrative approach to predicting the functional effects of non-coding and coding sequence variation," *Bioinformatics*, vol. 31, no. 10, pp. 1536–1543, May 2015. DOI: 10.1093/bioinformatics/btv009 (cit. on p. 33).

[173] X. Liu, C. Wu, C. Li, and E. Boerwinkle, "dbNSFP v3.0: A One-Stop Database of Functional Predictions and Annotations for Human Non-synonymous and Splice Site SNVs," *Hum Mutat*, vol. 37, no. 3, pp. 235–241, Mar. 2016. DOI: 10.1002/humu.22932 (cit. on p. 33).

[174] 1000 Genomes Project Consortium *et al.*, "A global reference for human genetic variation," *Nature*, vol. 526, no. 7571, pp. 68–74, Oct. 2015. DOI: 10.1038/nature15393 (cit. on p. 33).

[175] I. Ionita-Laza, K. McCallum, B. Xu, and J. D. Buxbaum, "A spectral approach integrating functional genomic annotations for coding and noncoding variants," *Nature Genetics*, vol. 48, no. 2, pp. 214–220, Feb. 2016. DOI: 10.1038/ng.3477 (cit. on p. 33).

[176] D. Smedley *et al.*, "A whole-genome analysis framework for effective identification of pathogenic regulatory variants in mendelian disease," *The American Journal of Human Genetics*, vol. 99, no. 3, pp. 595–606, Sep. 2016. DOI: 10.1016/j.ajhg.2016.07.005 (cit. on p. 33).

[177] Y.-F. Huang, B. Gulko, and A. Siepel, "Fast, scalable prediction of deleterious noncoding variants from functional and population genomic data," *Nature Genetics*, vol. 49, no. 4, pp. 618–624, Apr. 2017. DOI: 10.1038/ng.3810 (cit. on p. 33).

[178] I. F. A. C. Fokkema *et al.*, "Dutch genome diagnostic laboratories accelerated and improved variant interpretation and increased accuracy by sharing data," *Human Mutation*, vol. 40, no. 12, pp. 2230–2238, 2019. DOI: 10.1002/humu.23896 (cit. on p. 33).

[179] P. Rentzsch, M. Schubach, J. Shendure, and M. Kircher, "CADD-Splice—improving genome-wide variant effect prediction using deep learning-derived splice scores," *Genome Medicine*, vol. 13, no. 1, p. 31, Feb. 2021. DOI: 10.1186/s13073-021-00835-9 (cit. on p. 33).

[180] J. Shamsani *et al.*, "A plugin for the Ensembl Variant Effect Predictor that uses MaxEntScan to predict variant spliceogenicity," *Bioinformatics*, vol. 35, no. 13, pp. 2315–2317, Jul. 2019. DOI: 10.1093/bioinformatics/bty960 (cit. on p. 33).

[181] J. Wang, J. Zhang, K. Li, W. Zhao, and Q. Cui, "SpliceDisease database: Linking RNA splicing and disease," *Nucleic Acids Res*, vol. 40, no. Database issue, pp. D1055–1059, Jan. 2012. DOI: 10.1093/nar/gkr1171 (cit. on p. 33).

[182] S. Gelfman *et al.*, "Annotating pathogenic non-coding variants in genic regions," *Nat Commun*, vol. 8, no. 1, p. 236, Aug. 2017. DOI: 10.1038/s41467-017-00141-2 (cit. on pp. 34, 39).

[183] D. Danis *et al.*, "Interpretable prioritization of splice variants in diagnostic next-generation sequencing," *Am J Hum Genet*, vol. 108, no. 9, pp. 1564–1577, Sep. 2021. DOI: 10.1016/j.ajhg.2021.06.014 (cit. on pp. 34, 39).

[184] S. T. Sherry *et al.*, "dbSNP: The NCBI database of genetic variation," *Nucleic Acids Research*, vol. 29, no. 1, pp. 308–311, Jan. 2001. DOI: 10.1093/nar/29.1.308 (cit. on p. 34).

[185] J.-i. Takeda, S. Fukami, A. Tamura, A. Shibata, and K. Ohno, "IntSplice2: Prediction of the Splicing Effects of Intronic Single-Nucleotide Variants Using LightGBM Modeling," *Frontiers in Genetics*, vol. 12, 2021. DOI: 10.3389/fgene.2021.701076 (cit. on pp. 35, 44).

[186] R. Soemedi *et al.*, "Pathogenic variants that alter protein code often disrupt splicing," *Nat Genet*, vol. 49, no. 6, pp. 848–855, Jun. 2017. DOI: 10.1038/ng.3837 (cit. on p. 35).

[187] P. Barbosa, *Preparing input for multiple splicing predictors*, https://github.com/PedroBarbosa/Prepare_SplicingPredictors, 2023 (cit. on p. 35).

[188] W. McLaren *et al.*, "The Ensembl Variant Effect Predictor," *Genome Biology*, vol. 17, no. 1, p. 122, Jun. 2016. DOI: 10.1186/s13059-016-0974-4 (cit. on pp. 36, 151).

[189] Y. Bromberg, R. Prabakaran, A. Kabir, and A. Shehu, "Variant Effect Prediction in the Age of Machine Learning," *Cold Spring Harb Perspect Biol*, a041467, Apr. 2024. DOI: 10.1101/cshperspect.a041467 (cit. on p. 36).

[190] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011 (cit. on p. 37).

[191] D. G. Grimm *et al.*, "The evaluation of tools used to predict the impact of missense variants is hindered by two types of circularity.," *Human mutation*, vol. 36, no. 5, pp. 513–23, May 2015. DOI: 10.1002/humu.22768 (cit. on p. 38).

[192] H. Jung, K. S. Lee, and J. K. Choi, "Comprehensive characterisation of intronic mis-splicing mutations in human cancers," *Oncogene*, vol. 40, no. 7, pp. 1347–1361, Feb. 2021. DOI: 10.1038/s41388-020-01614-3 (cit. on pp. 42, 44).

[193] U. S. S. Petersen, T. K. Doktor, and B. S. Andresen, "Pseudoexon activation in disease by non-splice site deep intronic sequence variation — wild type pseudoexons constitute high-risk sites in the human genome," *Human Mutation*, vol. 43, no. 2, pp. 103–127, 2022. DOI: 10.1002/humu.24306 (cit. on pp. 44, 52, 153).

[194] S. I. Adamson, L. Zhan, and B. R. Graveley, "Vex-seq: High-throughput identification of the impact of genetic variation on pre-mRNA splicing efficiency," *Genome Biology*, vol. 19, no. 1, p. 71, Jun. 2018. DOI: 10.1186/s13059-018-1437-x (cit. on pp. 44, 153).

[195] R. Cheung *et al.*, "A Multiplexed Assay for Exon Recognition Reveals that an Unappreciated Fraction of Rare Genetic Variants Cause Large-Effect Splicing Disruptions," *Mol Cell*, vol. 73, no. 1, 183–194.e8, Jan. 2019. DOI: 10.1016/j.molcel.2018.10.037 (cit. on p. 44).

[196] R. Dawes *et al.*, "SpliceVault predicts the precise nature of variant-associated mis-splicing," *Nat Genet*, vol. 55, no. 2, pp. 324–332, Feb. 2023. DOI: 10.1038/s41588-022-01293-8 (cit. on p. 49).

[197] S. Köhler *et al.*, "The Human Phenotype Ontology in 2021," *Nucleic Acids Research*, vol. 49, no. D1, pp. D1207–D1217, Jan. 2021. DOI: 10.1093/nar/gkaa1043 (cit. on pp. 51, 157).

[198] S. Richards *et al.*, "Standards and Guidelines for the Interpretation of Sequence Variants: A Joint Consensus Recommendation of the American College of Medical Genetics and Genomics and the Association for Molecular Pathology," *Genet Med*, vol. 17, no. 5, pp. 405–424, May 2015. DOI: 10.1038/gim.2015.30 (cit. on p. 52).

[199] K. Schoch *et al.*, "Alternative transcripts in variant interpretation: The potential for missed diagnoses and misdiagnoses," *Genetics in Medicine*, vol. 22, no. 7, pp. 1269–1275, Jul. 2020. DOI: 10.1038/s41436-020-0781-x (cit. on p. 52).

[200] D. Canson, D. Glubb, and A. B. Spurdle, "Variant effect on splicing regulatory elements, branchpoint usage, and pseudoexonization: Strategies to enhance bioinformatic prediction using hereditary cancer genes as exemplars," *Human Mutation*, vol. 41, no. 10, pp. 1705–1721, 2020. DOI: 10.1002/humu.24074 (cit. on p. 53).

[201] L. Grodecká, E. Buratti, and T. Freiberger, "Mutations of Pre-mRNA Splicing Regulatory Elements: Are Predictions Moving Forward to Clinical Diagnostics?" *Int J Mol Sci*, vol. 18, no. 8, p. 1668, Jul. 2017. DOI: 10.3390/ijms18081668 (cit. on p. 53).

[202] F. Gebauer, T. Schwarzl, J. Valcárcel, and M. W. Hentze, "RNA-binding proteins in human genetic disease," *Nat Rev Genet*, vol. 22, no. 3, pp. 185–198, Mar. 2021. DOI: 10.1038/s41576-020-00302-y (cit. on pp. 53, 68).

[203] T. Ching *et al.*, "Opportunities and obstacles for deep learning in biology and medicine," *Journal of The Royal Society Interface*, vol. 15, no. 141, p. 20 170 387, Apr. 2018. DOI: 10.1098/rsif.2017.0387 (cit. on p. 53).

[204] J. K. Aicher, P. Jewell, J. Vaquero-Garcia, Y. Barash, and E. J. Bhoj, "Mapping RNA splicing variations in clinically-accessible and non-accessible tissues to facilitate Mendelian disease diagnosis using RNA-seq," *Genetics in medicine : official journal of the American College of Medical Genetics*, vol. 22, no. 7, p. 1181, Jul. 2020. DOI: 10.1038/s41436-020-0780-y (cit. on p. 53).

[205] C. Smith and J. O. Kitzman, "Benchmarking splice variant prediction algorithms using massively parallel splicing assays," *Genome Biology*, vol. 24, no. 1, p. 294, Dec. 2023. DOI: 10.1186/s13059-023-03144-z (cit. on p. 54).

[206] J.-M. de Sainte Agathe *et al.*, "SpliceAI-visual: A free online tool to improve SpliceAI splicing variant interpretation," *Hum Genomics*, vol. 17, p. 7, Feb. 2023. DOI: 10.1186/s40246-023-00451-1 (cit. on pp. 54, 89).

[207] T. Wolf *et al.*, *HuggingFace's Transformers: State-of-the-art Natural Language Processing*, Jul. 2020. DOI: 10.48550/arXiv.1910.03771. arXiv: 1910.03771 [cs] (cit. on p. 55).

[208] Ž. Avsec *et al.*, "Effective gene expression prediction from sequence by integrating long-range interactions," *Nat Methods*, vol. 18, no. 10, pp. 1196–1203, Oct. 2021. DOI: 10.1038/s41592-021-01252-x (cit. on p. 55).

[209] J. Meier, R. Rao, R. Verkuil, J. Liu, T. Sercu, and A. Rives, "Language models enable zero-shot prediction of the effects of mutations on protein function," in *Advances in Neural Information Processing Systems*, vol. 34, Curran Associates, Inc., 2021, pp. 29 287–29 303 (cit. on p. 55).

[210] A. Frankish *et al.*, "GENCODE 2021," *Nucleic Acids Research*, vol. 49, no. D1, pp. D916–D923, Jan. 2021. DOI: 10.1093/nar/gkaa1087 (cit. on p. 58).

[211] M. Chen and J. L. Manley, "Mechanisms of alternative splicing regulation: Insights from molecular and genomics approaches," *Nat Rev Mol Cell Biol*, vol. 10, no. 11, pp. 741–754, Nov. 2009. DOI: 10.1038/nrm2777 (cit. on pp. 60, 80).

[212] P. Sheng, K. A. Flood, and M. Xie, "Short Hairpin RNAs for Strand-Specific Small Interfering RNA Production," *Front. Bioeng. Biotechnol.*, vol. 8, Aug. 2020. DOI: 10.3389/fbioe.2020.00940 (cit. on p. 60).

[213] C. Y. Chan *et al.*, "A structural interpretation of the effect of GC-content on efficiency of RNA interference," *BMC Bioinformatics*, vol. 10, no. Suppl 1, S33, Jan. 2009. DOI: 10.1186/1471-2105-10-S1-S33 (cit. on p. 60).

[214] M. Amit *et al.*, "Differential GC content between exons and introns establishes distinct strategies of splice-site recognition," *Cell Rep*, vol. 1, no. 5, pp. 543–556, May 2012. DOI: 10.1016/j.celrep.2012.03.013 (cit. on p. 60).

[215] C. E. Grant, T. L. Bailey, and W. S. Noble, "FIMO: Scanning for occurrences of a given motif," *Bioinformatics*, vol. 27, no. 7, pp. 1017–1018, Apr. 2011. DOI: 10.1093/bioinformatics/btr064 (cit. on pp. 65, 68, 97).

[216] G. Giudice, F. Sánchez-Cabo, C. Torroja, and E. Lara-Pezzi, "ATtRACT—a database of RNA-binding proteins and associated motifs," *Database*, vol. 2016, baw035, Jan. 2016. DOI: 10.1093/database/baw035 (cit. on pp. 65, 66).

[217] L. P. Benoit Bouvrette, S. Bovaird, M. Blanchette, and E. Lécuyer, "oRNAment: A database of putative RNA binding protein target sites in the transcriptomes of model species," *Nucleic Acids Research*, vol. 48, no. D1, pp. D166–D173, Jan. 2020. DOI: 10.1093/nar/gkz986 (cit. on pp. 65, 97).

[218] L. E. Marasco and A. R. Kornblihtt, "The physiology of alternative splicing," *Nat Rev Mol Cell Biol*, Oct. 2022. DOI: 10.1038/s41580-022-00545-z (cit. on pp. 66, 67).

[219] W. S. Noble, "How does multiple testing correction work?" *Nat Biotechnol*, vol. 27, no. 12, pp. 1135–1137, Dec. 2009. DOI: 10.1038/nbt1209-1135 (cit. on p. 68).

[220] T. Achsel and C. Bagni, "Cooperativity in RNA–protein interactions: The complex is more than the sum of its partners," *Current Opinion in Neurobiology*, Cellular Neuroscience, vol. 39, pp. 146–151, Aug. 2016. DOI: 10.1016/j.conb.2016.06.007 (cit. on p. 68).

[221] X. Li, G. Quon, H. D. Lipshitz, and Q. Morris, "Predicting in vivo binding sites of RNA-binding proteins using mRNA secondary structure," *RNA*, vol. 16, no. 6, pp. 1096–1107, Jun. 2010. DOI: 10.1261/rna.2017210 (cit. on p. 68).

[222] D. Dominguez *et al.*, "Sequence, Structure, and Context Preferences of Human RNA Binding Proteins," *Molecular Cell*, vol. 70, no. 5, 854–867.e9, Jun. 2018. DOI: 10.1016/j.molcel.2018.05.001 (cit. on pp. 68, 79).

[223] T. Saldi, K. Riemondy, B. Erickson, and D. L. Bentley, "Alternative RNA structures formed during transcription depend on elongation rate and modify RNA processing," *Molecular Cell*, vol. 81, no. 8, 1789–1801.e5, Apr. 2021. DOI: 10.1016/j.molcel.2021.01.040 (cit. on p. 68).

[224] J. Imig, A. Kanitz, and A. P. Gerber, "RNA regulons and the RNA-protein interaction network," *Biomol Concepts*, vol. 3, no. 5, pp. 403–414, Oct. 2012. DOI: 10.1515/bmc-2012-0016 (cit. on p. 68).

[225] J. J. D. Ho *et al.*, "A network of RNA-binding proteins controls translation efficiency to activate anaerobic metabolism," *Nat Commun*, vol. 11, p. 2677, May 2020. DOI: 10.1038/s41467-020-16504-1 (cit. on p. 68).

[226] J. Han *et al.*, "SR Proteins Induce Alternative Exon Skipping through Their Activities on the Flanking Constitutive Exons," *Mol Cell Biol*, vol. 31, no. 4, pp. 793–802, Feb. 2011. DOI: 10.1128/MCB.01117-10 (cit. on p. 70).

[227] S. C. Huelga *et al.*, "Integrative genome-wide analysis reveals cooperative regulation of alternative splicing by hnRNP proteins," *Cell Rep*, vol. 1, no. 2, pp. 167–178, Feb. 2012. DOI: 10.1016/j.celrep.2012.02.001 (cit. on p. 71).

[228] S. Pandit *et al.*, "Genome-wide Analysis Reveals SR Protein Cooperation and Competition in Regulated Splicing," *Molecular Cell*, vol. 50, no. 2, pp. 223–235, Apr. 2013. DOI: 10.1016/j.molcel.2013.03.001 (cit. on pp. 72, 74).

[229] J. M. Howard and J. R. Sanford, "THE RNAissance Family: SR proteins as multifaceted regulators of gene expression," *Wiley Interdiscip Rev RNA*, vol. 6, no. 1, pp. 93–110, Jan. 2015. DOI: 10.1002/wrna.1260 (cit. on p. 72).

[230] J. Ule *et al.*, "An RNA map predicting Nova-dependent splicing regulation," *Nature*, vol. 444, no. 7119, pp. 580–586, Nov. 2006. DOI: 10.1038/nature05304 (cit. on p. 72).

[231] S. Ishigaki *et al.*, "Position-dependent FUS-RNA interactions regulate alternative splicing events and transcriptions," *Sci Rep*, vol. 2, no. 1, p. 529, Jul. 2012. DOI: 10.1038/srep00529 (cit. on p. 75).

[232] M. Mikl, A. Hamburg, Y. Pilpel, and E. Segal, "Dissecting splicing decisions and cell-to-cell variability with designed sequence libraries," *Nat Commun*, vol. 10, p. 4572, Oct. 2019. DOI: 10.1038/s41467-019-12642-3 (cit. on p. 75).

[233] M. Ghandi, D. Lee, M. Mohammad-Noori, and M. A. Beer, "Enhanced Regulatory Sequence Prediction Using Gapped k-mer Features," *PLoS Comput Biol*, vol. 10, no. 7, e1003711, Jul. 2014. DOI: 10.1371/journal.pcbi.1003711 (cit. on p. 76).

[234] D. Lee, "LS-GKM: A new gkm-SVM for large-scale datasets," *Bioinformatics*, vol. 32, no. 14, pp. 2196–2198, Jul. 2016. DOI: 10.1093/bioinformatics/btw142 (cit. on p. 76).

[235] A. Shrikumar, *Lsgkm+gkmexplain with regression functionality*, Zenodo, Dec. 2020. DOI: 10.5281/zenodo.4300866 (cit. on p. 76).

[236] D. Ray *et al.*, "RNA-binding proteins that lack canonical RNA-binding domains are rarely sequence-specific," *Sci Rep*, vol. 13, no. 1, p. 5238, Mar. 2023. DOI: 10.1038/s41598-023-32245-9 (cit. on p. 78).

[237] M. W. Hentze, A. Castello, T. Schwarzl, and T. Preiss, "A brave new world of RNA-binding proteins," *Nat Rev Mol Cell Biol*, vol. 19, no. 5, pp. 327–341, May 2018. DOI: 10.1038/nrm.2017.130 (cit. on p. 78).

[238] P. Julien, B. Miñana, P. Baeza-Centurion, J. Valcárcel, and B. Lehner, "The complete local genotype-phenotype landscape for the alternative splicing of a human exon," *Nat Commun*, vol. 7, p. 11 558, May 2016. DOI: 10.1038/ncomms11558 (cit. on pp. 80, 89, 116).

[239] A. Penev, A. Bazley, M. Shen, J. D. Boeke, S. A. Savage, and A. Sfeir, "Alternative splicing is a developmental switch for hTERT expression," *Molecular Cell*, vol. 81, no. 11, 2349–2360.e6, Jun. 2021. DOI: 10.1016/j.molcel.2021.03.033 (cit. on p. 80).

[240] E. I. Prakash, A. Shrikumar, and A. Kundaje, "Towards More Realistic Simulated Datasets for Benchmarking Deep Learning Models in Regulatory Genomics," in *Proceedings of the 16th Machine Learning in Computational Biology Meeting*, PMLR, Jan. 2022, pp. 58–77 (cit. on pp. 82, 83).

[241] J. Zrimec *et al.*, "Controlling gene expression with deep generative design of regulatory DNA," *Nat Commun*, vol. 13, no. 1, p. 5099, Aug. 2022. DOI: 10.1038/s41467-022-32818-8 (cit. on p. 83).

[242] S. M. Castillo-Hair and G. Seelig, "Machine Learning for Designing Next-Generation mRNA Therapeutics," *Acc. Chem. Res.*, vol. 55, no. 1, pp. 24–34, Jan. 2022. DOI: 10.1021/acs.accounts.1c00621 (cit. on p. 83).

[243] N. Killoran, L. J. Lee, A. Delong, D. Duvenaud, and B. J. Frey, *Generating and designing DNA with deep generative models*, Dec. 2017. DOI: 10.48550/arXiv.1712.06148 (cit. on p. 83).

[244] J. Linder, N. Bogard, A. B. Rosenberg, and G. Seelig, "A Generative Neural Network for Maximizing Fitness and Diversity of Synthetic DNA and Protein Sequences," *Cell Systems*, vol. 11, no. 1, 49–62.e16, Jul. 2020. DOI: 10.1016/j.cels.2020.05.007 (cit. on p. 83).

[245] D. Brookes, H. Park, and J. Listgarten, "Conditioning by adaptive sampling for robust design," in *Proceedings of the 36th International Conference on Machine Learning*, PMLR, May 2019, pp. 773–782 (cit. on p. 83).

[246] J. Linder and G. Seelig, "Fast activation maximization for molecular sequence design," *BMC Bioinformatics*, vol. 22, no. 1, p. 510, Oct. 2021. DOI: 10.1186/s12859-021-04437-5 (cit. on p. 83).

[247] J. A. Valeri *et al.*, "BioAutoMATED: An end-to-end automated machine learning tool for explanation and design of biological sequences," *Cell Systems*, vol. 14, no. 6, 525–542.e9, Jun. 2023. DOI: 10.1016/j.cels.2023.05.007 (cit. on p. 83).

[248] I. J. Goodfellow *et al.*, *Generative Adversarial Networks*, Jun. 2014. DOI: 10.48550/arXiv.1406.2661. arXiv: 1406.2661 (cit. on p. 83).

[249] D. P. Kingma and M. Welling, *Auto-Encoding Variational Bayes*, Dec. 2022. DOI: 10.48550/arXiv.1312.6114 (cit. on p. 83).

[250] S. Sinai, R. Wang, A. Whatley, S. Slocum, E. Locane, and E. D. Kelsic, *AdaLead: A simple and robust adaptive greedy search algorithm for sequence design*, Oct. 2020. DOI: 10.48550/arXiv.2010.02141 (cit. on p. 83).

[251] C. Angermueller *et al.*, "Population-Based Black-Box Optimization for Biological Sequence Design," in *Proceedings of the 37th International Conference on Machine Learning*, PMLR, Nov. 2020, pp. 324–334 (cit. on p. 83).

[252] A. Klie *et al.*, "Predictive analyses of regulatory sequences with EUGENe," *Nat Comput Sci*, vol. 3, no. 11, pp. 946–956, Nov. 2023. DOI: 10.1038/s43588-023-00544-w (cit. on p. 83).

[253] I. I. Taskiran *et al.*, "Cell type directed design of synthetic enhancers," *Nature*, Dec. 2023. DOI: 10.1038/s41586-023-06936-2 (cit. on pp. 83, 90).

[254] N. K. Lee, Z. Tang, S. Toneyan, and P. K. Koo, "EvoAug: Improving generalization and interpretability of genomic deep neural networks with evolution-inspired data augmentations," *Genome Biology*, vol. 24, no. 1, p. 105, May 2023. DOI: 10.1186/s13059-023-02941-w (cit. on p. 83).

[255] R. García-Pérez *et al.*, "The landscape of expression and alternative splicing variation across human traits," *Cell Genom*, vol. 3, no. 1, p. 100 244, Jan. 2023. DOI: 10.1016/j.xgen.2022.100244 (cit. on p. 89).

[256] I. Cascino, G. Fiucci, G. Papoff, and G. Ruberti, "Three functional soluble forms of the human apoptosis-inducing Fas molecule are produced by alternative splicing," *J Immunol*, vol. 154, no. 6, pp. 2706–2713, Mar. 1995 (cit. on p. 89).

[257] J. M. Izquierdo *et al.*, "Regulation of Fas Alternative Splicing by Antagonistic Effects of TIA-1 and PTB on Exon Definition," *Molecular Cell*, vol. 19, no. 4, pp. 475–484, Aug. 2005. DOI: 10.1016/j.molcel.2005.06.015 (cit. on p. 89).

[258] P. Baeza-Centurion, B. Miñana, J. M. Schmiedel, J. Valcárcel, and B. Lehner, "Combinatorial Genetics Reveals a Scaling Law for the Effects of Mutations on Splicing," *Cell*, vol. 176, no. 3, 549–563.e23, Jan. 2019. DOI: 10.1016/j.cell.2018.12.010 (cit. on pp. 89, 116).

[259] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, *Optuna: A Next-generation Hyperparameter Optimization Framework*, Jul. 2019. DOI: 10.48550/arXiv.1907.10902. arXiv: 1907.10902 (cit. on p. 90).

[260] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., vol. 24, Curran Associates, Inc., 2011 (cit. on p. 90).

[261] L. Spector, "Assessment of problem modality by differential performance of lexicase selection in genetic programming: A preliminary report," in *Proceedings of the 14th Annual Conference Companion on Genetic and Evolutionary Computation*, ser. GECCO '12, New York, NY, USA: Association for Computing Machinery, Jul. 2012, pp. 401–408. DOI: 10.1145/2330784.2330846 (cit. on p. 93).

[262] L. T. Gehman *et al.*, "The splicing regulator Rbfox2 is required for both cerebellar development and mature motor function," *Genes Dev*, vol. 26, no. 5, pp. 445–460, Mar. 2012. DOI: 10.1101/gad.182477.111 (cit. on p. 95).

[263] J. P. Venables *et al.*, "RBFOX2 Is an Important Regulator of Mesenchymal Tissue-Specific Splicing in both Normal and Cancer Tissues," *Mol Cell Biol*, vol. 33, no. 2, pp. 396–405, Jan. 2013. DOI: 10.1128/MCB.01174-12 (cit. on p. 95).

[264] A. Jbara *et al.*, "RBFOX2 modulates a metastatic signature of alternative splicing in pancreatic cancer," *Nature*, pp. 1–7, Mar. 2023. DOI: 10.1038/s41586-023-05820-3 (cit. on p. 95).

[265] P. Baeza-Centurion, B. Miñana, J. Valcárcel, and B. Lehner, "Mutations primarily alter the inclusion of alternatively spliced exons," *eLife*, vol. 9, C. P. Ponting, P. J. Wittkopp, and N. L. Barbosa-Morais, Eds., e59959, Oct. 2020. DOI: 10.7554/eLife.59959 (cit. on p. 96).

[266] I. Agarkova, D. Auerbach, E. Ehler, and J. C. Perriard, "A novel marker for vertebrate embryonic heart, the EH-myomesin isoform," *J Biol Chem*, vol. 275, no. 14, pp. 10 256–10 264, Apr. 2000. DOI: 10.1074/jbc.275.14.10256 (cit. on p. 105).

[267] R. Schoenauer *et al.*, "Myomesin is a molecular spring with adaptable elasticity," *J Mol Biol*, vol. 349, no. 2, pp. 367–379, Jun. 2005. DOI: 10.1016/j.jmb.2005.03.055 (cit. on p. 105).

[268] Ž. Avsec, M. Barekatain, J. Cheng, and J. Gagneur, "Modeling positional effects of regulatory sequences with spline transformations increases prediction accuracy of deep neural networks," *Bioinformatics*, vol. 34, no. 8, pp. 1261–1269, Apr. 2018. DOI: 10.1093/bioinformatics/btx727 (cit. on p. 115).

[269] M. Ghanbari and U. Ohler, "Deep neural networks for interpreting RNA-binding protein target preferences," *Genome Res.*, vol. 30, no. 2, pp. 214–226, Feb. 2020. DOI: 10.1101/gr.247494.118 (cit. on p. 115).

[270] M. Horlacher *et al.*, "Towards in silico CLIP-seq: Predicting protein-RNA interaction via sequence-to-signal learning," *Genome Biology*, vol. 24, no. 1, p. 180, Aug. 2023. DOI: 10.1186/s13059-023-03015-7 (cit. on p. 116).

[271] B. J. Livesey and J. A. Marsh, "Using deep mutational scanning to benchmark variant effect predictors and identify disease mutations," *Molecular Systems Biology*, vol. 16, no. 7, e9380, Jul. 2020. DOI: 10.15252/msb.20199380 (cit. on p. 116).

[272] S. Braun *et al.*, "Decoding a cancer-relevant splicing decision in the RON proto-oncogene using high-throughput mutagenesis," *Nat Commun*, vol. 9, no. 1, p. 3315, Aug. 2018. DOI: 10.1038/s41467-018-05748-7 (cit. on p. 116).

[273] P. Gergics *et al.*, "High-throughput splicing assays identify missense and silent splice-disruptive POU1F1 variants underlying pituitary hormone deficiency," *Am J Hum Genet*, vol. 108, no. 8, pp. 1526–1539, Aug. 2021. DOI: 10.1016/j.ajhg.2021.06.013 (cit. on p. 116).

[274] C. Smith, B. B. Burugula, I. Dunn, S. Aradhya, J. O. Kitzman, and J. L. Yee, "High-Throughput Splicing Assays Identify Known and Novel WT1 Exon 9 Variants in Nephrotic Syndrome," *Kidney Int Rep*, vol. 8, no. 10, pp. 2117–2125, Oct. 2023. DOI: 10.1016/j.ekir.2023.07.033 (cit. on p. 116).

[275] K. Gupta *et al.*, "Improved modeling of RNA-binding protein motifs in an interpretable neural model of RNA splicing," *Genome Biology*, vol. 25, no. 1, p. 23, Jan. 2024. DOI: 10.1186/s13059-023-03162-x (cit. on p. 117).

[276] J. T. Witten and J. Ule, "Understanding splicing regulation through RNA splicing maps," *Trends Genet*, vol. 27, no. 3-2, pp. 89–97, Mar. 2011. DOI: 10.1016/j.tig.2010.12.001 (cit. on p. 118).

[277] D. Vitsios, R. S. Dhindsa, L. Middleton, A. B. Gussow, and S. Petrovski, "Prioritizing non-coding regions based on human genomic constraint and sequence context with deep learning," *Nat Commun*, vol. 12, no. 1, p. 1504, Mar. 2021. DOI: 10.1038/s41467-021-21790-4 (cit. on p. 118).

[278] M. Yuksekgonul, M. Wang, and J. Zou, *Post-hoc Concept Bottleneck Models*, Feb. 2023. DOI: 10.48550/arXiv.2205.15480. arXiv: 2205.15480 [cs, stat] (cit. on p. 118).

[279] OpenAI *et al.*, *GPT-4 Technical Report*, Mar. 2024. DOI: 10.48550/arXiv.2303.08774. arXiv: 2303.08774 [cs] (cit. on p. 118).

[280] A. Rives *et al.*, "Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences," *Proceedings of the National Academy of Sciences*, vol. 118, no. 15, e2016239118, Apr. 2021. DOI: 10.1073/pnas.2016239118 (cit. on p. 118).

[281] J. A. Ruffolo and A. Madani, "Designing proteins with language models," *Nat Biotechnol*, vol. 42, no. 2, pp. 200–202, Feb. 2024. DOI: 10.1038/s41587-024-02123-4 (cit. on p. 118).

[282] H. Dalla-Torre *et al.*, *The Nucleotide Transformer: Building and Evaluating Robust Foundation Models for Human Genomics*, Sep. 2023. DOI: 10.1101/2023.01.11.523679 (cit. on p. 118).

[283] E. Nguyen *et al.*, *HyenaDNA: Long-Range Genomic Sequence Modeling at Single Nucleotide Resolution*, Nov. 2023. DOI: 10.48550/arXiv.2306.15794. arXiv: 2306.15794 [cs, q-bio] (cit. on p. 118).

[284] V. Fishman *et al.*, *GENA-LM: A Family of Open-Source Foundational DNA Language Models for Long Sequences*, Nov. 2023. DOI: 10.1101/2023.06.12.544594 (cit. on p. 118).

[285] Z. Zhou, Y. Ji, W. Li, P. Dutta, R. Davuluri, and H. Liu, *DNABERT-2: Efficient Foundation Model and Benchmark For Multi-Species Genome*, Mar. 2024. DOI: 10.48550/arXiv.2306.15006. arXiv: 2306.15006 [cs, q-bio] (cit. on p. 118).

[286] Y. Ji, Z. Zhou, H. Liu, and R. V. Davuluri, "DNABERT: Pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome," *Bioinformatics*, vol. 37, no. 15, pp. 2112–2120, Aug. 2021. DOI: 10.1093/bioinformatics/btab083 (cit. on p. 118).

[287] G. Benegas, S. S. Batra, and Y. S. Song, "DNA language models are powerful predictors of genome-wide variant effects," *Proceedings of the National Academy of Sciences*, vol. 120, no. 44, e2311219120, Oct. 2023. DOI: 10.1073/pnas.2311219120 (cit. on p. 118).

[288] E. Nguyen *et al.*, *Sequence modeling and design from molecular to genome scale with Evo*, Mar. 2024. DOI: 10.1101/2024.02.27.582234 (cit. on p. 118).

[289] M. Poli *et al.*, *Hyena Hierarchy: Towards Larger Convolutional Language Models*, Apr. 2023. DOI: 10.48550/arXiv.2302.10866. arXiv: 2302.10866 [cs] (cit. on p. 118).

[290] Z. Tang and P. K. Koo, *Evaluating the representational power of pre-trained DNA language models for regulatory genomics*, Mar. 2024. DOI: 10.1101/2024.02.29.582810 (cit. on p. 119).

[291] K. Chen, Y. Zhou, M. Ding, Y. Wang, Z. Ren, and Y. Yang, "Self-supervised learning on millions of primary RNA sequences from 72 vertebrates improves sequence-based RNA splicing prediction," *Briefings in Bioinformatics*, vol. 25, no. 3, bbae163, May 2024. DOI: 10.1093/bib/bbae163 (cit. on p. 119).

[292] A. Celaj *et al.*, *An RNA foundation model enables discovery of disease mechanisms and candidate therapeutics*, Sep. 2023. DOI: 10.1101/2023.09.20.558508 (cit. on p. 119).

[293] J. Linder, D. Srivastava, H. Yuan, V. Agarwal, and D. R. Kelley, *Predicting RNA-seq coverage from DNA sequence as a unifying model of gene regulation*, Sep. 2023. DOI: 10.1101/2023.08.30.555582 (cit. on p. 119).

[294] B. P. de Almeida *et al.*, *SegmentNT: Annotating the genome at single-nucleotide resolution with DNA foundation models*, Mar. 2024. DOI: 10.1101/2024.03.14.584712 (cit. on p. 119).

[295] O. Ronneberger, P. Fischer, and T. Brox, *U-Net: Convolutional Networks for Biomedical Image Segmentation*, May 2015. DOI: 10.48550/arXiv.1505.04597. arXiv: 1505.04597 [cs] (cit. on p. 119).

[296] S. Vilov and M. Heinig, *Investigating the performance of foundation models on human 3'UTR sequences*, Feb. 2024. DOI: 10.1101/2024.02.09.579631 (cit. on p. 120).

[297] K. Dudnyk, D. Cai, C. Shi, J. Xu, and J. Zhou, "Sequence basis of transcription initiation in the human genome," *Science*, vol. 384, no. 6694, eadj0116, Apr. 2024. DOI: 10.1126/science.adj0116 (cit. on p. 120).

[298] S. Naqvi *et al.*, *Transfer learning reveals sequence determinants of the quantitative response to transcription factor dosage*, May 2024. DOI: 10.1101/2024.05.28.596078 (cit. on p. 120).

[299] C. Wilks, P. Gaddipati, A. Nellore, and B. Langmead, "Snaptron: Querying splicing patterns across tens of thousands of RNA-seq samples," *Bioinformatics*, vol. 34, no. 1, pp. 114–116, Jan. 2018. DOI: 10.1093/bioinformatics/btx547 (cit. on p. 153).

[300] X. Liu, C. Li, C. Mou, Y. Dong, and Y. Tu, "dbNSFP v4: A comprehensive database of transcript-specific functional predictions and annotations for human nonsynonymous and splice-site SNVs," *Genome Medicine*, vol. 12, no. 1, p. 103, Dec. 2020. DOI: 10.1186/s13073-020-00803-9 (cit. on p. 154).

[301] W. J. Kent *et al.*, "The Human Genome Browser at UCSC," *Genome Res.*, vol. 12, no. 6, pp. 996–1006, Jun. 2002. DOI: 10.1101/gr.229102 (cit. on p. 154).

[302] B. S. Pedersen, R. M. Layer, and A. R. Quinlan, "Vcfanno: Fast, flexible annotation of genetic variants," *Genome Biology*, vol. 17, no. 1, p. 118, Jun. 2016. DOI: 10.1186/s13059-016-0973-5 (cit. on p. 154).

[303] A. C. Gomes, P. S. Barbosa, A. Coutinho, I. Cruz, M. Carmo-Fonseca, and L. R. Lopes, "Whole-genome DNA sequencing: The key to detecting a sarcomeric mutation in a 'false genotype-negative' family with hypertrophic cardiomyopathy," *Revista Portuguesa de Cardiologia*, vol. 39, no. 4, 227.e1–227.e9, Apr. 2020. DOI: 10.1016/j.repc.2019.03.011 (cit. on p. 177).

[304] S. Matos *et al.*, "Screening a Targeted Panel of Genes by Next-Generation Sequencing Improves Risk Stratification in Real World Patients with Acute Myeloid Leukemia," *Cancers*, vol. 14, no. 13, p. 3236, Jun. 2022. DOI: 10.3390/cancers14133236 (cit. on p. 177).

[305] A. Usié *et al.*, "An improved reference genome and first organelle genomes of Quercus suber," *Tree Genetics & Genomes*, vol. 19, no. 6, p. 54, Nov. 2023. DOI: 10.1007/s11295-023-01624-8 (cit. on p. 178).

[306] A. C. Raposo *et al.*, *Erosion of X-Chromosome Inactivation in female hiPSCs is heterogeneous and persists during differentiation*, Mar. 2024. DOI: 10.1101/2024.03.15.585169 (cit. on p. 178).

Appendix A

Predicting intronic variants affecting the splicing mechanism

A.1 Data collection

We employed the same variant annotation procedure for all the variants collected for this study (datasets described below). We used Ensembl VEP v109 [188] for the task and transcript annotations were added accordingly (with `--per_gene --pick_order ccds,canonical,biotype, rank --no_intergenic --gencode_basic` set).

ClinVar

We downloaded ClinVar v202204 and selected all the SNV for downstream analysis. We kept variants with Pathogenic and Benign assignments (`CLNSIG` == Pathogenic or Likely_pathogenic or Benign or Likely_benign). We identified intronic variants based on Ensembl VEP annotations: only variants with at least one intronic consequence (`INTRON` == 1) in a protein-coding transcript (`BIOTYPE` = protein_coding) were retained. Additionally, we excluded variants with exonic annotations in any other gene (`EXON` $\neq$ 1). To avoid being overly conservative, we added variants that Ensembl VEP annotated as being outside the gene body for the picked consequence (`Consequence` == TF_binding_site_variant or downstream_gene_variant or upstream_gene_variant or regulatory_region_variant), but that are annotated with intronic ontology terms in the `MC` field in the original VCF. To minimize labeling errors, we excluded variants with less than one confidence star. To ensure that the number of benign variants did not exceed 50,000 (and therefore avoid the dataset being excessively unbalanced), we selected all higher-confidence benign variants (with two or more stars, N=13,093) along with 36,907 randomly chosen one-star variants. Finally, we retrieved the RefSeq transcript ID associated with each variant and selected only those that were intronic in

such reference transcript. The dataset size for raw evaluations amounted to 18,446 pathogenic and 49,343 benign variants.

Disease-causing intronic variants affecting RNA splicing

This dataset refers to a high-quality variant set that we carefully curated to comply with the following criteria:

- Variant must locate at more than 10bp from the nearest splice site.
- Variant was experimentally proven to affect normal RNA splicing.
- Variant does not necessarily lead to pseudoexon activation.

The previous curation effort from our lab [59] was updated for this manuscript to include a comprehensive set of intronic variants identified after 2017. Therefore, the positive (disease-causing) set of variants used in this benchmark totals 242 (81 Vaz-Drago *et al.* [59] and 161 from the new curation effort).

We used gnomAD v2.1 to generate a matched control set. First, we extracted all gnomAD variants occurring in a window of 500bp surrounding the variants in the positive set and selected common records with a frequency higher 0.01 (1%) in the population, resulting in 1128 variants. Then, we ran Ensembl VEP as previously described and retained the intronic variants annotated as occurring in one of the 148 unique genes of the positive set (N=1091). Moreover, we kept variants absent in ClinVar having the VCF filter field as `PASS` (N=546). Finally, we randomly sampled 242 from this set.

Variants that affect RNA splicing

The third main dataset refers to variants that affect different mechanisms of splicing regulation, which may or may not lead to disease. We defined different molecular categories based on the location of the variant relative to the abnormal splicing event. We focused on deep intronic variants that lead to partial intron retention or pseudoexon activation. In cases where a variant leads to both pseudoexon activation and partial intron retention, we have assigned it to the pseudoexon activation group. Exceptionally, we included variants that affect the branchpoint motif (thus, closer to annotated splice acceptors) that include other types of splicing alterations such as exon skipping. We defined each category as follows:

- *Branchpoint associated*, for those variants occurring between -18 and 44bp upstream (as used in [151]) of an annotated or cryptic splicing acceptor site, and that create or disrupt any of the following adenine-branchpoint consensus motifs: YTNAY, YTNA, TNA, YNA [131].
- *Acceptor Upstream*, referring to any variant that locates between -2 and -18bp upstream of the cryptic splice acceptor (including the polypyrimidine tract).

- *New Splice Acceptor*, denoting the variants that occur at the cryptic splice acceptor positions, including the first nucleotide of the cryptic exon.
- *Exonic-like*, for any variant occurring within the cryptic exon (pseudoexon or partially retained intron).
- *New Splice Donor*, composed of variants located at the cryptic splice donor positions, including the last position of the cryptic exon.
- *Donor Downstream*, referring to any deep intronic variant that locates at a distance of more than 2bp from the activated cryptic splice donor.

We used data produced or gathered from multiple studies to assign variants to each category (Table 4.2). While splicing-altering variants were straightforward to assign (based on source data, the functional consequence, and distances to the splicing element considered), we distributed the non-altering variants such that they resembled as best as possible the spatial distribution of the positive sets. Hence, we assigned the negative variants taking into account two levels of information: the primary group (partial intron retention, pseudoexon activation) and the region category.

To keep in line with the expected biology, we assigned the variants that were within a defined distance to a splice site to the partial intron retention group and deeper intronic variants to the pseudoexon group. We used different distance thresholds for splice acceptors and donors (100bp and 20bp, respectively) so that the datasets were reasonably balanced. As for the region category, we defined negative variants occurring between 18 and 44bp upstream of an annotated splicing acceptor as branchpoint-associated variants. We assigned as acceptor-upstream or donor-downstream the remaining intronic variants according to whether they were located upstream or downstream of the nearest annotated splice site. Because pseudoexons tend to resemble authentic exons [193], we exceptionally assigned exonic variants that did not change inclusion levels of tested exons [194] as controls for the Exonic-like category.

Lastly, we generated control datasets for the new splice site categories. Splice site variants (located at one of the dinucleotide positions) that were experimentally tested to not affect splicing are not easily accessible. Therefore, to mimic the positive set, we looked for common deep intronic SNV ($>$ 5% in gnomAD v2.1) in protein-coding transcripts that generate the most common 5-mer acceptor motif CAGGT in the human genome [58] through a mutation in the core splice site dinucleotide. We randomly selected 64 variants to match the number of positive new splice acceptor variants exactly. We employed the same procedure for the new splice donor variants, where we kept the variants that generate the most common 6-mer donor motif GGTAAG in the human genome [58] at the GT position. Finally, we selected 197 variants at random to match the number of positive new splice donor variants. We confirmed using Snaptron [299] that in GTEx data, there is no evidence that a splice junction is used at the variant intervals.

A.2 Prediction tools

We selected an extensive list of prediction tools for evaluation. The single criterion for the inclusion of a tool was that it had to be designed to predict (at least partially) intronic variation. When available, we used pre-computed scores to annotate our variant sets (from dbNSFP v4.0b1 [300], UCSC genome browser [301], Zenodo or tool website). Otherwise, we ran the models directly following the developer's instructions. For a subset of splicing-related tools (MMSplice, HAL, kipoiSplice4), we employed kipoi v0.8.6 [82] to get predictions. We additionally included splicing-related tools that predict specific splicing signals (e.g., BP) and are not necessarily targeted to predict pathogenicity. Because most of these tools do not score variants by design, and some require using a web-based portal, we developed a simple utility to prepare their input given a VCF file. Moreover, we created a script for each tool to process the raw output into a final prediction score to be included in a VCF file. The package is available at `https://github.com/PedroBarbosa/Prepare_SplicingPredictors`. We annotated the final VCF files with all the predictions using vcfanno v0.3.3 [302]. We describe all the tools, their reference thresholds and how we ran them in Table 4.1.

A.3 Performance evaluation

We used VETA v0.7.8 [36] to perform all the performance evaluations. In this study, we employed different metrics according to the nature of the dataset and the goal of the analysis. For ClinVar data, we ranked variants based on the F1-score, given the unbalanced nature of the data (much more deep intronic benign variants than pathogenic). Because some tools do not score deep in the introns (missing data), we weighted the F1-score with the prediction coverage: $Coverage \cdot \left(2 \cdot \frac{(Precision \cdot Recall)}{(Precision + Recall)}\right)$ where $Coverage = \frac{Scored\_variants}{Total\_variants}$. For balanced datasets, we ranked tools using a slight variation of the Matthews Correlation Coefficient (MCC) ($MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$) that normalizes the metric range between 0 and 1 ($normalizedMCC = \frac{MCC+1}{2}$). We weighted the normalized MCC values with the prediction coverage ($weighted\_normalized\_MCC = Coverage \cdot normalizedMCC$). Additionally, we employed ROC and Precision-Recall Curves (PR Curves) for the comparisons that measure performance at multiple threshold values. To summarize such analyses, we used the auROC and the auPRC metrics, respectively.

A.4 Further inspection of deep intronic variants in ClinVar

We selected ClinVar variants assigned to the "501-1000" and "+1000" intronic bins and used ensembl VEP to perform reannotation. We ran ensembl VEP using RefSeq annotations without picking any consequence (`--per_gene` and `--pick_order` were not set), meaning that all

transcript consequences associated with each variant were retained. We employed a filter to only keep annotations of protein-coding transcripts. Then, we assigned each variant to one of four categories, according to the overlap configuration of transcripts belonging to the gene associated with the variant: if a variant is exonic in another overlapping transcript, we termed it as "Exonic"; if a variant is located at a shorter distance from the splice site in any other transcript, we assigned the category "> 1 transcript (smaller offset)"; if the distance to the closest splice site remains the same for all transcripts overlapping the variant, we assigned the variant to the "> 1 transcript (smaller offset)" category; lastly, if no other transcript overlapped with the variant (besides the one used in the analysis), we set it to the "No other transcript" category.

A.5 Assessing quality of interpretations for SPiP, SQURLS and SpliceVault

For this task, we employed the dataset of pathogenic splicing variants used throughout the study. It includes variants from our curation plus variants from Vaz-Drago *et al.* [59] because the molecular mechanism for the splicing defect is known for almost all records (Supplementary Table S3). For SPiP and SQUIRLS, we ran VETA in the `interrogate` mode (with `--labels Pathogenic` set) to list the variants correctly predicted by each tool using the threshold calibrated for non-canonical intronic variation (SPiP > 0.009 and SQUIRLS > 0.016). We removed variants for which the ground truth information was not available (e.g.,pseudoexon-activating variants that lack details of the location of the variant concerning the cryptic event).

For SPiP, we parsed the output so that the interpretation tag, confidence interval and original score were retrieved (3rd, 4th and 5th fields after splitting predictions by "|"). We assigned variants with an "NTR" tag (low probability of affecting splicing, yet correctly predicted as pathogenic according to the calibrated threshold) to the "No interpretation" category. Variants not associated with any particular splicing mechanism (according to SPiP, the "Alter by complex event" tag) were given the "Not informative" interpretation category. Then, for each of the remaining SPiP tags we classified the interpretation as correct if they matched the ground truth information:

- "Alter BP" for variants associated with the branchpoint signal, else incorrect.
- "Alter by create new Exon" for variants that trigger pseudoexon activation, else incorrect.
- "Alter by create New splice site" for variants that create a new splice site or activate a nearby existing cryptic splice site, regardless of the variant leading to pseudoexon activation or partial intron retention, else incorrect.
- "Alter ESR" for intronic variants occurring within the boundaries of a new pseudoexon,

else incorrect.
- "Alter by MES (Poly TC)" for polypyrimidine tract variants, else incorrect.

As for SQUIRLS, we ran the model for the subset of pathogenic variants correctly predicted by the tool using "–output-format html" and "–n-variants-to-report 121". Afterwards, we manually inspected the HTML report generated to derive structured interpretations for each variant: "Not informative" if the short description of the variant effect was not generated; "No interpretation" if SQUIRLS did not produce any description or figure for the variant; "New cryptic acceptor" and "New cryptic donor" if SQUIRLS described the creation of a new splice site and the variant was located at one of the splice site positions (based on the Sequence trekker figure) defined in this manuscript; "Activate cryptic acceptor" and "Activate cryptic donor" if SQUIRLS described the creation of a cryptic splice site and the variant was located outside of the splice site positions (based on the Sequence trekker figure). Because SQUIRLS does not predict the exact molecular effect of a splicing variant, we ignored the predicted number of bases affecting the coding sequence as this was not applicable for pseudoexon-activating variants. After manually inspecting the HTML report and generating structured interpretations, we classified the interpretation as correct if it matched the ground truth information:

- "New splice acceptor" for variants that create a new splice donor, else incorrect.
- "New splice donor" for variants that create a new splice acceptor, else incorrect.
- "Activate cryptic acceptor" for variants located upstream of an existing cryptic splice acceptor and not associated with the branchpoint signal, else incorrect.
- "Activate cryptic donor" for variants located downstream of an existing cryptic splice donor, else incorrect.

Finally, for SpliceVault we did not run a model to get correctly predicted variants to further inspect. Rather, SpliceVault is a web portal https://kidsneuro.shinyapps.io/splicevault/ (last accessed May 21st, 2023) to query non-canonical splicing patterns in large-scale population-based RNA-sequencing data. Because it relies on querying rare mis-splicing events with respect to annotated exons, we excluded all variants that trigger pseudoexon activation, as SpliceVault can't identify this class of events. As a result, 37 variants were left for evaluation. For each variant, we used the associated gene, intron number and molecular effect to select the correct exon and splice site to look for. We used the hg38 version (300k-RNA) and changed the default SpliceVault settings so that the Top-10 events per query were shown. Moreover, we allowed for all cryptic events to be reported, regardless of their distance to the target exon. We assigned variants to the "No interpretation" tag if the cryptic splicing event was not observed in SpliceVault Top-10 events. Then, we classified the interpretation as "Correct" if any of the cryptic splicing triggered by the variant was observed within the Top-4 events. This threshold was recommended by the authors of SpliceVault for clinical purposes. Conversely, if the event appeared in lower ranks, we classified the interpretation as "Incorrect".

A.6 Tissue-specific predictions by AbSplice-DNA

Throughout the manuscript, we selected the maximum AbSplice-DNA prediction for any tissue to evaluate model performance. In contrast, for this analysis, we used all predictions so that tissue specificity could be addressed. We used the same dataset as for the interpretability section. We ran VETA in the `interrogate` mode (with `--labels Pathogenic` set) to list the variants correctly predicted by AbSplice-DNA using the threshold adjusted for non-canonical intronic variation (>0.004, in at least one tissue). Then, for each variant, we gathered information about the tissues associated with the disease by searching the HPO [197] with the given OMIM disease identifier. We strived to assign tissue names that matched the GTEx tissues used by AbSplice-DNA. Disease-causing variants affecting tissues not represented in GTEx (e.g. retina) were discarded. Additionally, variants causing systemic diseases (e.g. Marfan syndrome), or diseases returning ambiguous HPO terms were excluded.

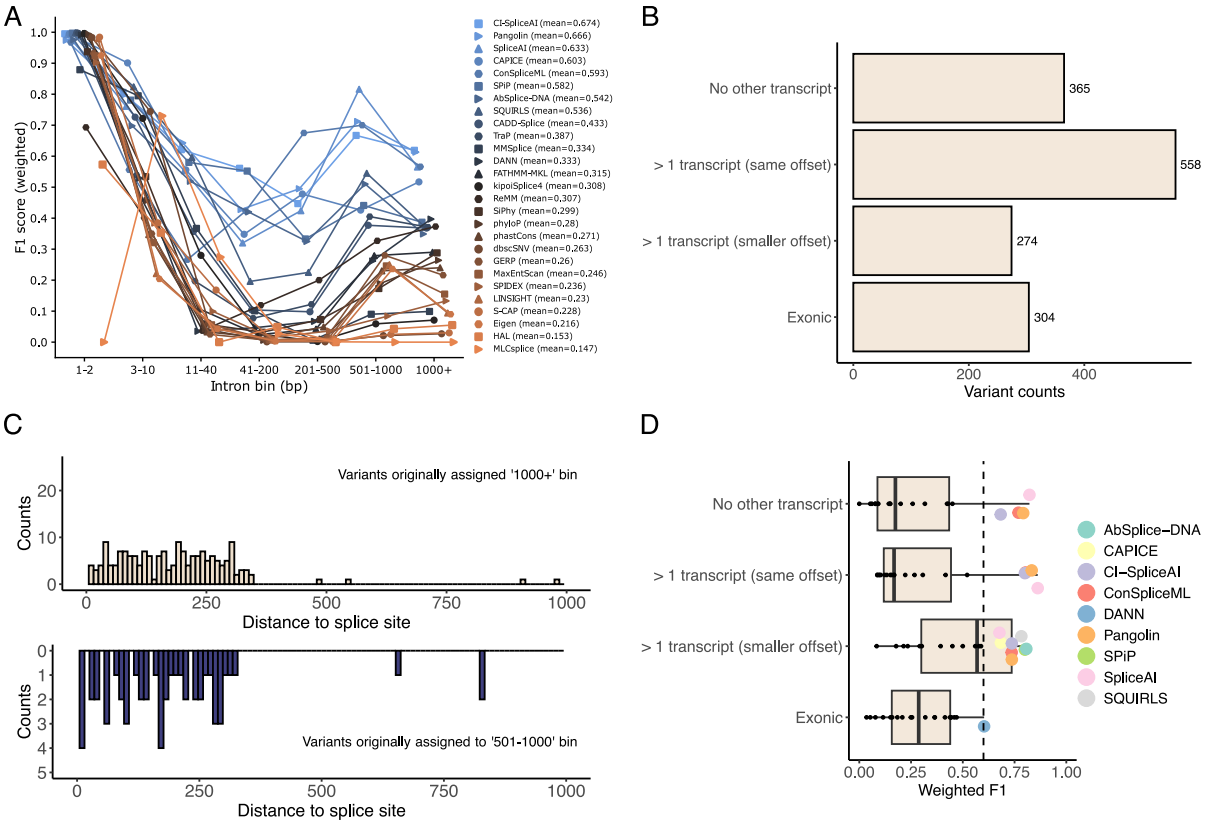

Figure A.1: Intronic variant prediction in ClinVar. **A** - Performance of all tools considered for the study on the raw ClinVar intronic dataset. Mean values in the legend represent the average weighted F1 score across all intronic bins. **B** - Inspection of intronic variants (after addressing circularity problems) assigned to the "501-1000" and "1000+" intronic bins. The bars reflect the number of variants assigned to each category. The term "No other transcript" refers to all variants that do not have any other RefSeq protein coding transcript of the same gene overlapping with them, besides the transcript originally used (N pathogenic=25, N benign=340). "> 1 transcript (same offset)" refers to variants that overlap with more than one transcript of the same gene but do not have any other transcript where the variant is closer to the splice site than in the original transcript used in the analysis (N pathogenic=31, N benign=527). "> 1 transcript (smaller offset)" refers to variants that overlap with more than one transcript of the same gene, and have at least one other transcript in which the variant is closer to the splice site than in the original transcript used in the analysis (N pathogenic=20, N benign=254). "Exonic" refers to variants that overlap with more than one transcript of the same gene, and have at least one other transcript where the variant is exonic (N pathogenic=41, N benign=263). **C** - Distribution of the updated intronic distances to the closest splice site for variants assigned to the "> 1 transcript (smaller offset)" category. **D** - Tool performance (measured with weighted F1 score) for each individual category. Tools with performance higher than 0.6 are highlighted.

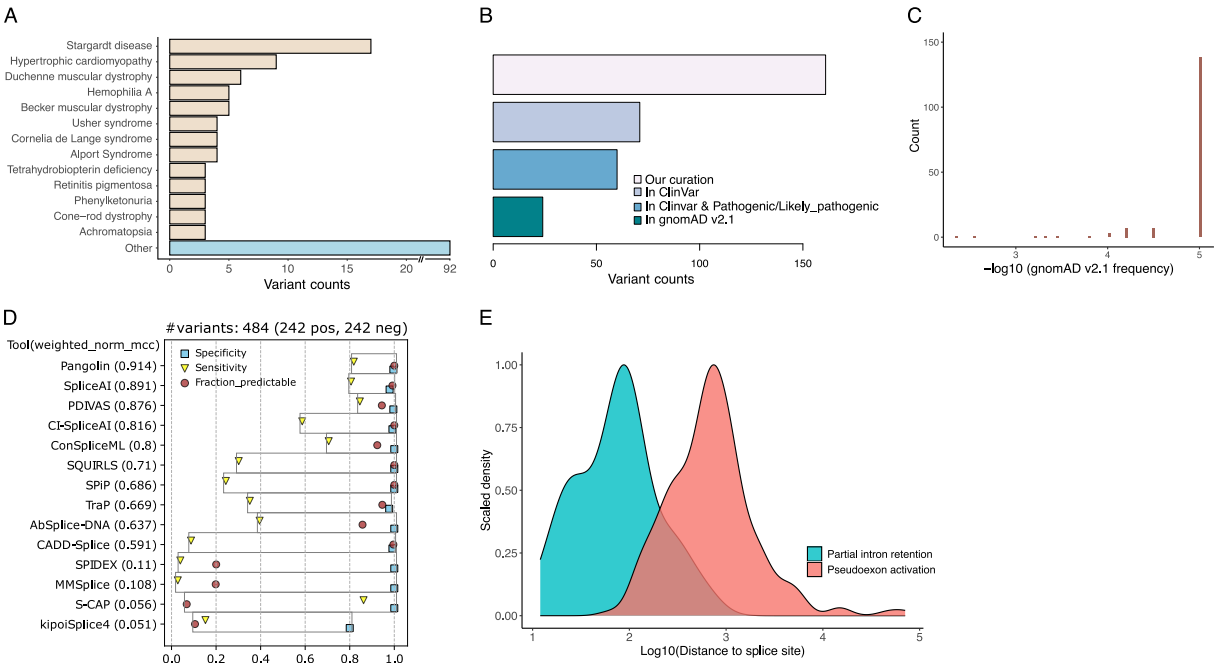

Figure A.2: Manually curated dataset of pathogenic intronic variants disrupting RNA splicing. **A** - Number of variants collected per phenotype. Diseases with less than 3 variants were assigned to the "Other" category. **B** - Number of variants occurring in ClinVar and gnomAD v2.1. **C** - Log transformed allele frequencies of variants in gnomAD v2.1. For those that are absent in the database, a pseudocount of 0.00001 was added (highest histogram peak, at 5). **D** - Tool performance using reference thresholds from Table 1 for variants curated in this manuscript plus those curated by Vaz-Drago *et al.* [59]. MLCsplice and dbscSNV are not shown as they had more than 95% of missing predictions. **E** - Distance (Log10) of the variants to the closest splice junction. Pseudoexon activation group: 194 variants; Partial intron retention group: 37 variants.

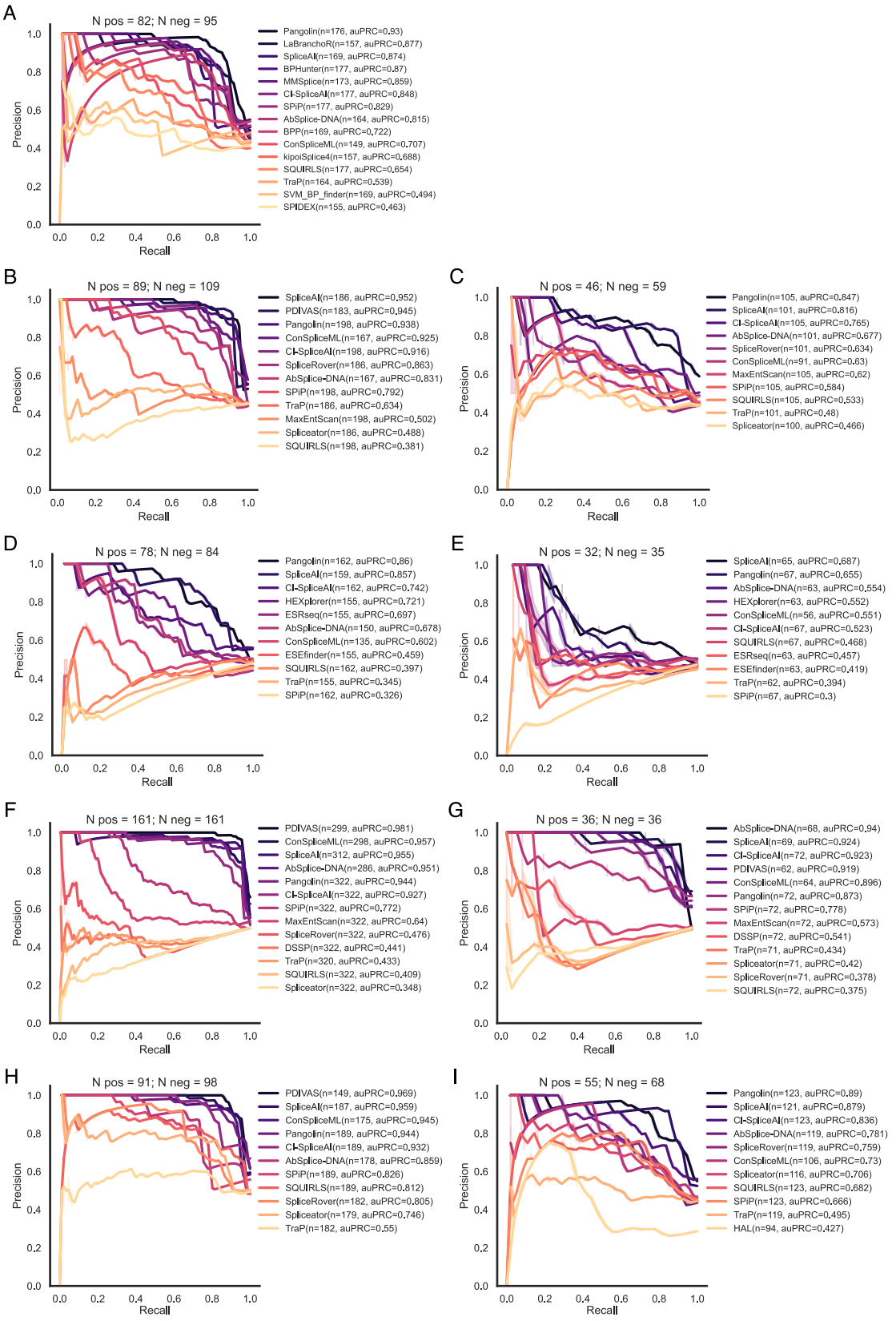

Figure A.3: Precision-Recall curves for all splicing-altering variants analyzed in a region-specific manner. Tools are ranked by the auPRC metric and the number of predictions made by each tool is displayed in `n=`. The number of variants in each dataset is presented ("N pos" and "N neg" represent the number of positive and negative splicing variants, respectively). Tools with more than 50% of missing predictions or with less than 15 variants in the minority class were excluded from these analyses. **A** - Branchpoint associated variants. **B** - Acceptor-associated variants triggering pseudoexon inclusion. **C** - Acceptor-associated variants leading to partial intron retention. **D** - Exonic-like variants triggering pseudoexon inclusion. **E** - Exonic-like variants leading to partial intron retention. **F** - Variants creating new splice donors and activate pseudoexons. **G** - Variants creating new splice donors and leading to partial intron retention. **H** - Variants activating existing upstream cryptic splice donors and triggering pseudoexon activation. **I** - Variants activating existing upstream cryptic splice donors and leading to partial intron retention.

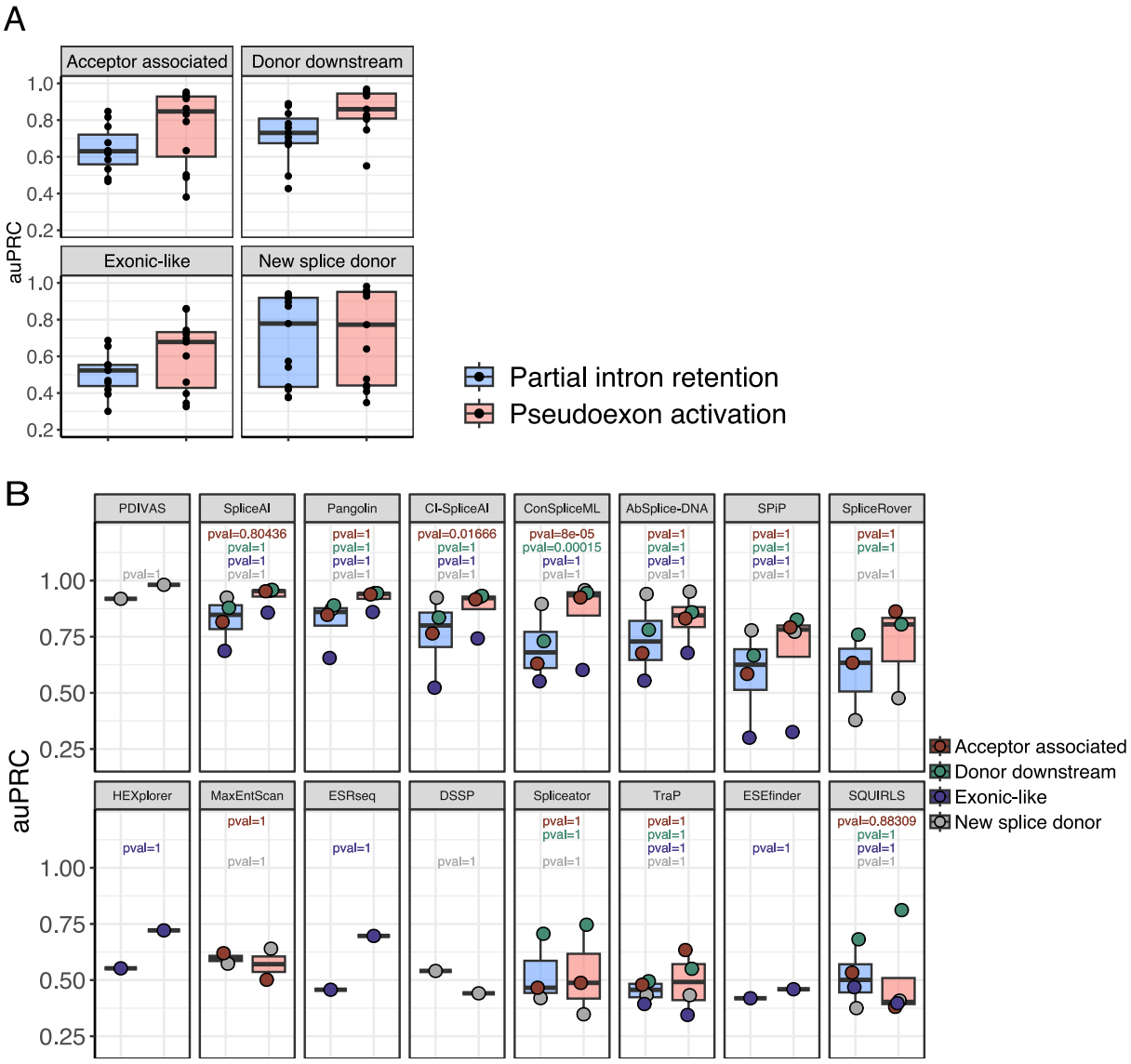

Figure A.4: Performance comparison between all pseudoexon activation versus partial intron retention variants. **A** - Distribution of the auPRC scores of the tools for each variant region. **B** - Per-tool auPRC distribution across the four variant regions considered. A Fisher's exact test was conducted separately for each tool and variant region to determine statistical significance for the performance differences between the pseudoexon activation and partial intron retention groups. The true positives plus true negatives were considered successful outcomes, while false positives plus false negatives were considered failures. The p-values displayed in the figure were corrected for multiple comparisons using the Holm method. For each tool, we excluded the variant regions that did not have performance measurements in both groups.

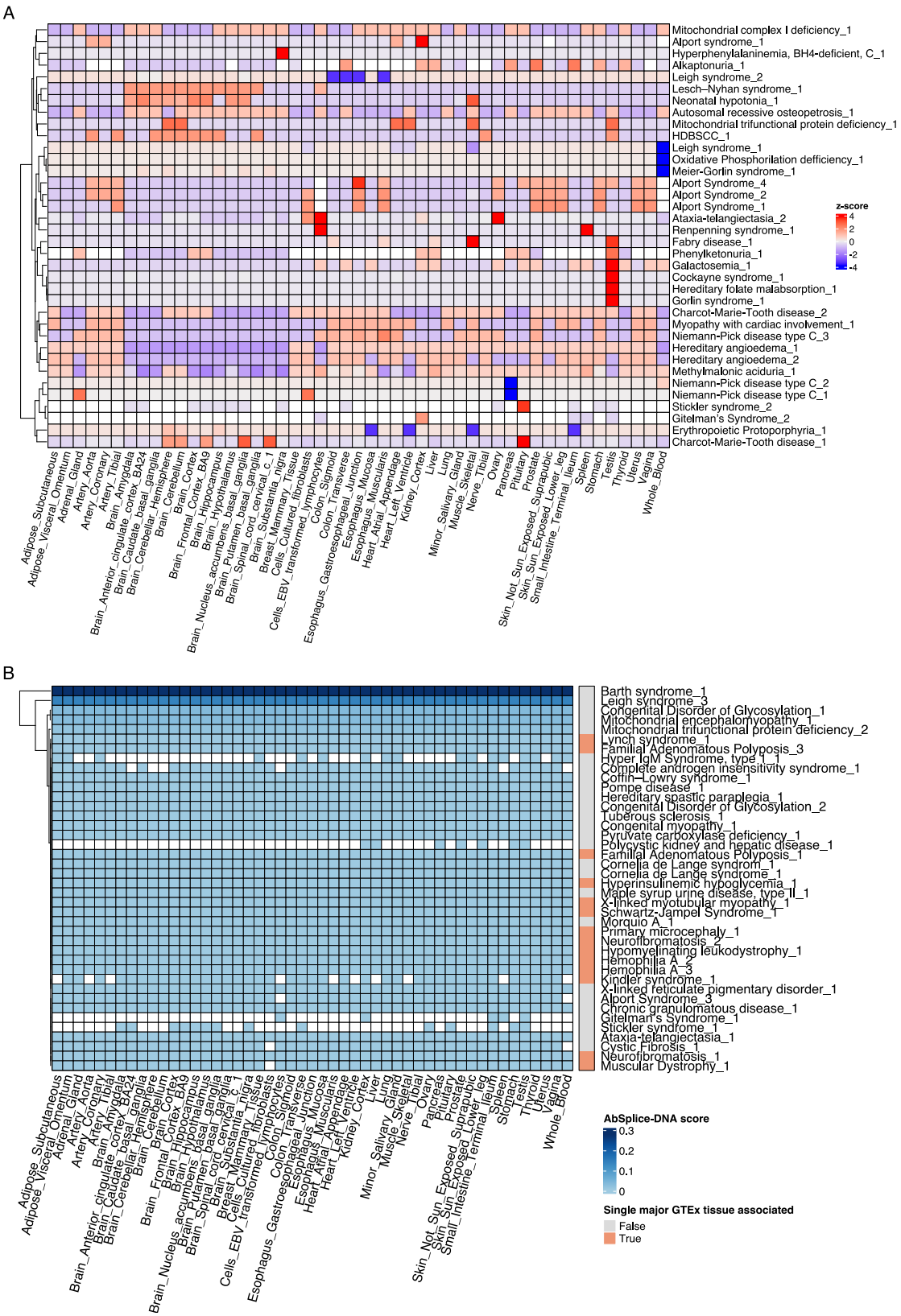

Figure A.5: Tissue-specific predictions made by AbSplice-DNA for a set of disease-causing variants associated with aberrant splicing. **A** - Disease variants associated with multiple GTEx tissues that displayed variable scores across tissues. **B** - Disease variants with no tissue-specificity. All tissues got the same AbSplice-DNA score. Disease variants associated with one or more GTEx tissues are displayed in a single heatmap annotation.

Table A.1: Adjusted thresholds that maximize performance for non-canonical intronic splicing variants at different levels of importance given to precision and recall based on the Fβ score, as described in Section 4.3.1.

Tool	Original threshold	Adjusted: β=0.5	Adjusted: β=1*	Adjusted: β=1.5
CADD-Splice	15	8.054	2.765	1.846
SQUIRLS	0.074	0.016	0.016	0.006
SpliceAI	0.2	0.1	0.05	0.04
TraP	0.289	0.211	0.062	0.0
ConSpliceML	0.5	0.19	0.07	0.03
CI-SpliceAI	0.19	0.08	0.02	0.02
Pangolin	0.2	0.158	0.053	0.053
SPiP	0.452	0.009	0.009	0.009
PDIVAS	0.151	0.03	0.01	0.01
AbSplice-DNA	0.01	0.004	0.004	0.0

* Thresholds selected for downstream analysis that used a binary decision threshold.

Appendix B

Interpreting SpliceAI

Figure B.1: Distribution of GC content values for upstream introns in each paired dataset. The `N=` at each facet (a single RBP paired dataset) indicates the total dataset size, including knockdown-sensitive and control exons.

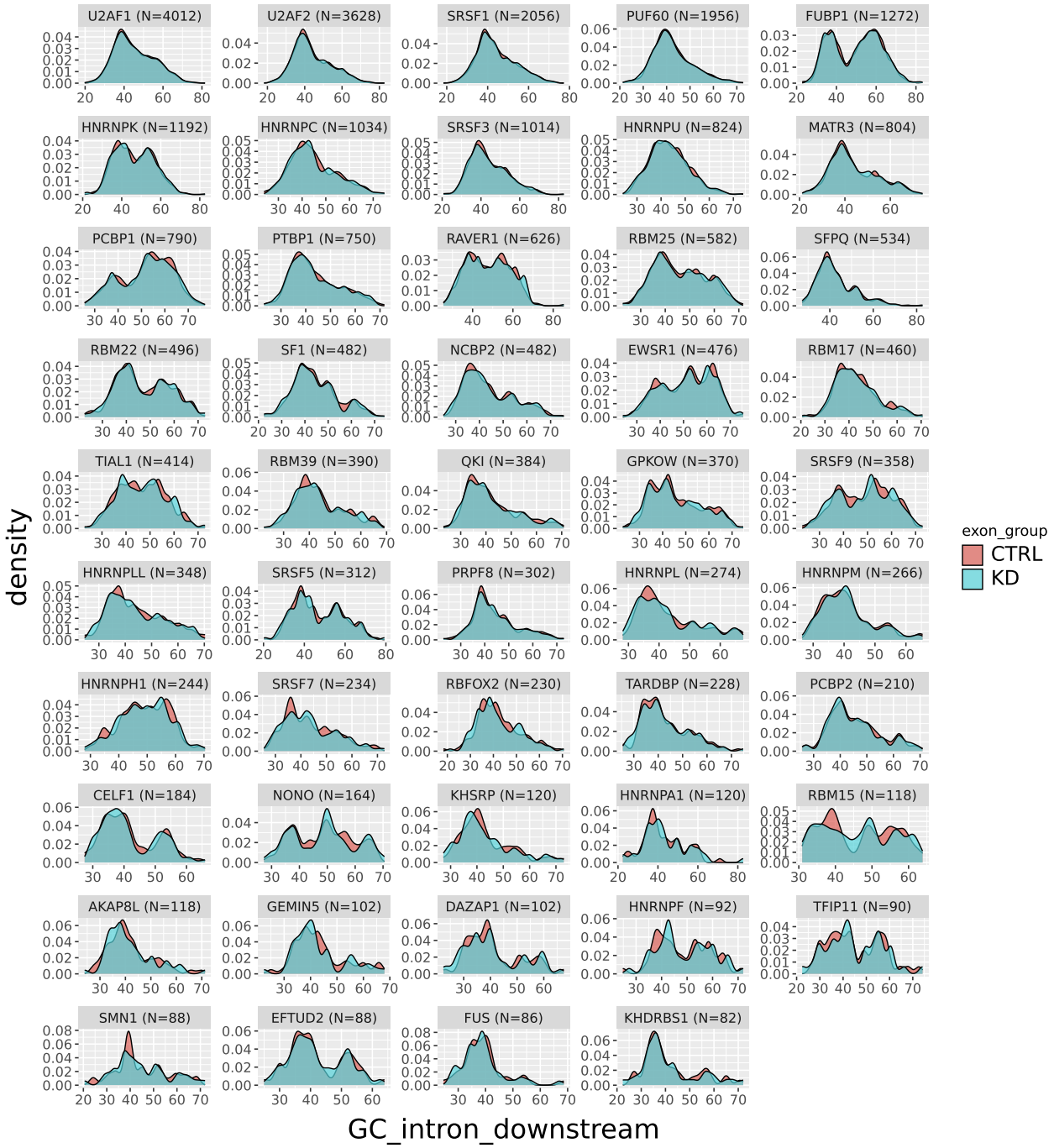

Figure B.2: Distribution of GC content values for downstream introns in each paired dataset. The `N=` at each facet (a single RBP paired dataset) indicates the total dataset size, including knockdown-sensitive and control exons.

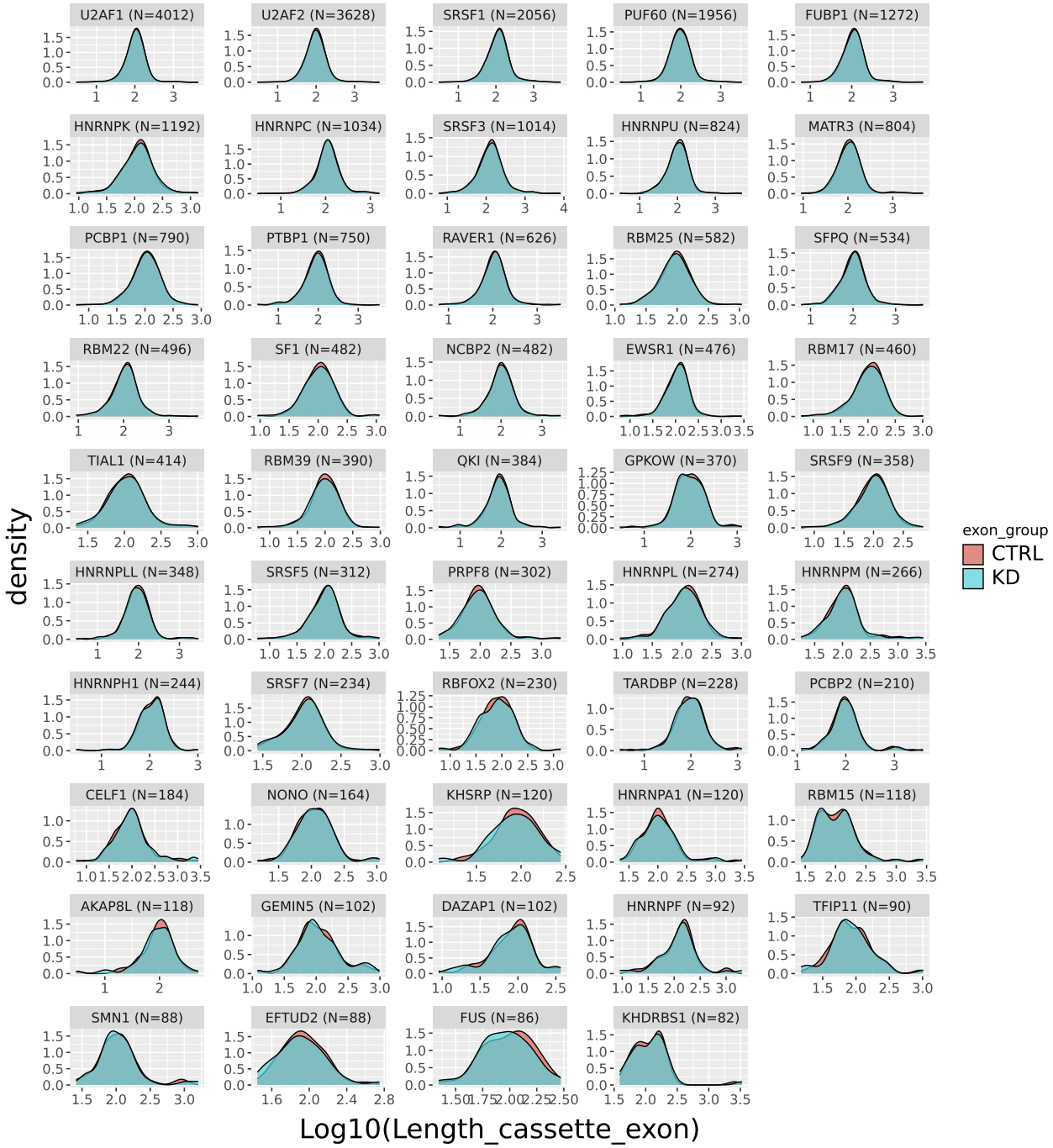

Figure B.3: Distribution of length values for cassette exons in each paired dataset. The `N=` at each facet (a single RBP paired dataset) indicates the total dataset size, including knockdown-sensitive and control exons.

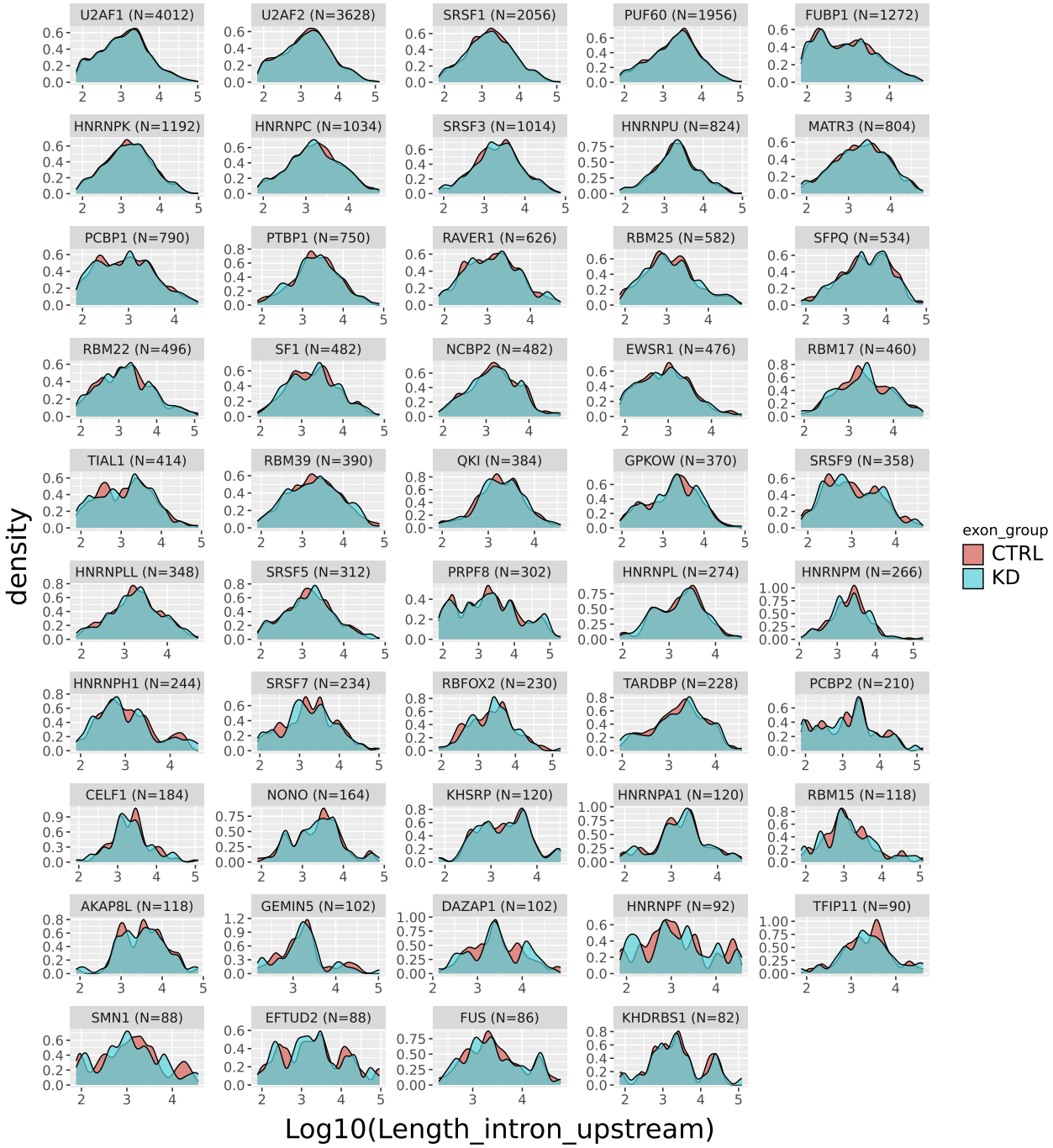

Figure B.4: Distribution of length values for upstream introns in each paired dataset. The `N=` at each facet (a single RBP paired dataset) indicates the total dataset size, including knockdown-sensitive and control exons.

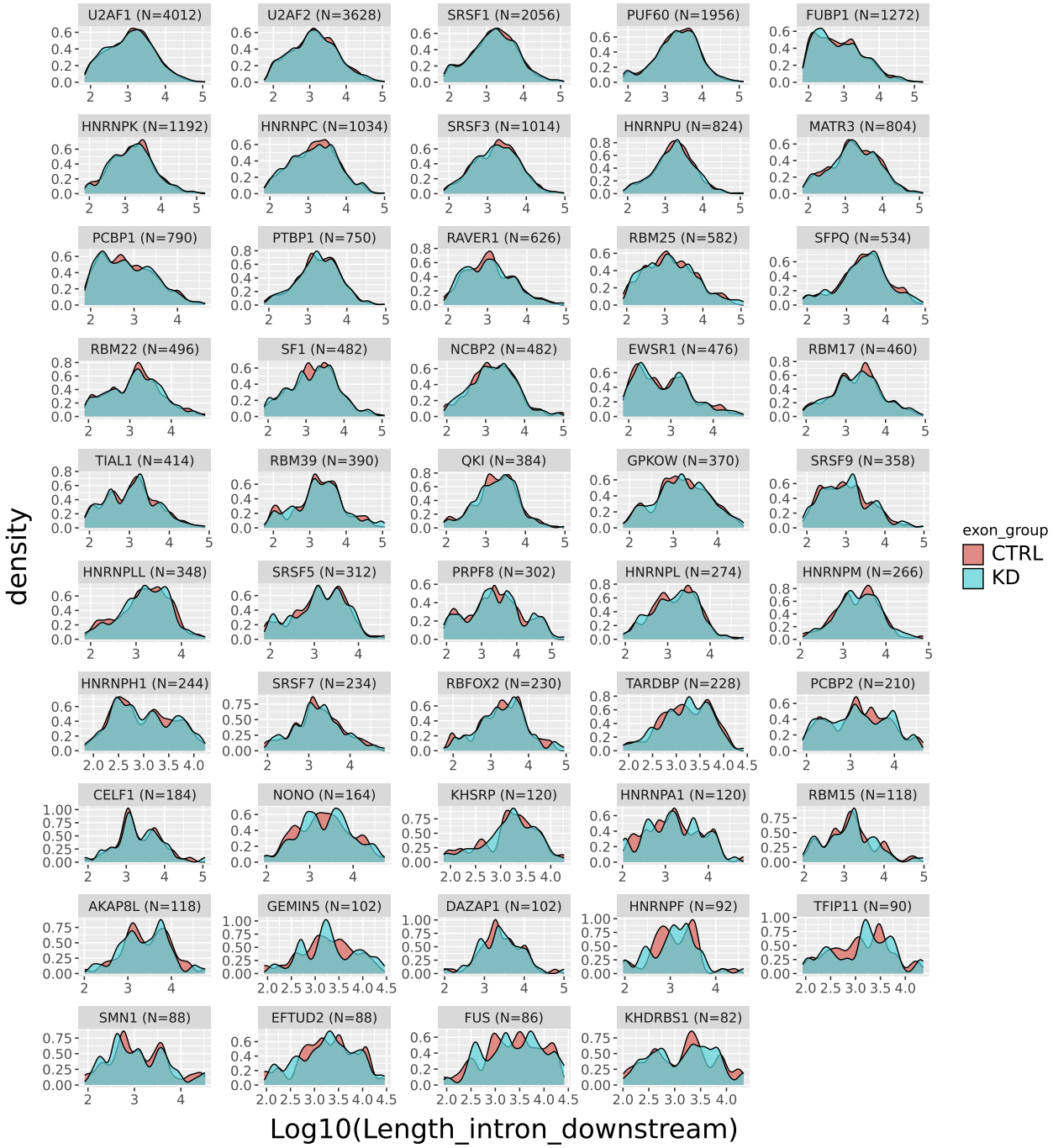

Figure B.5: Distribution of length values for downstream introns in each paired dataset. The `N=` at each facet (a single RBP paired dataset) indicates the total dataset size, including knockdown-sensitive and control exons.

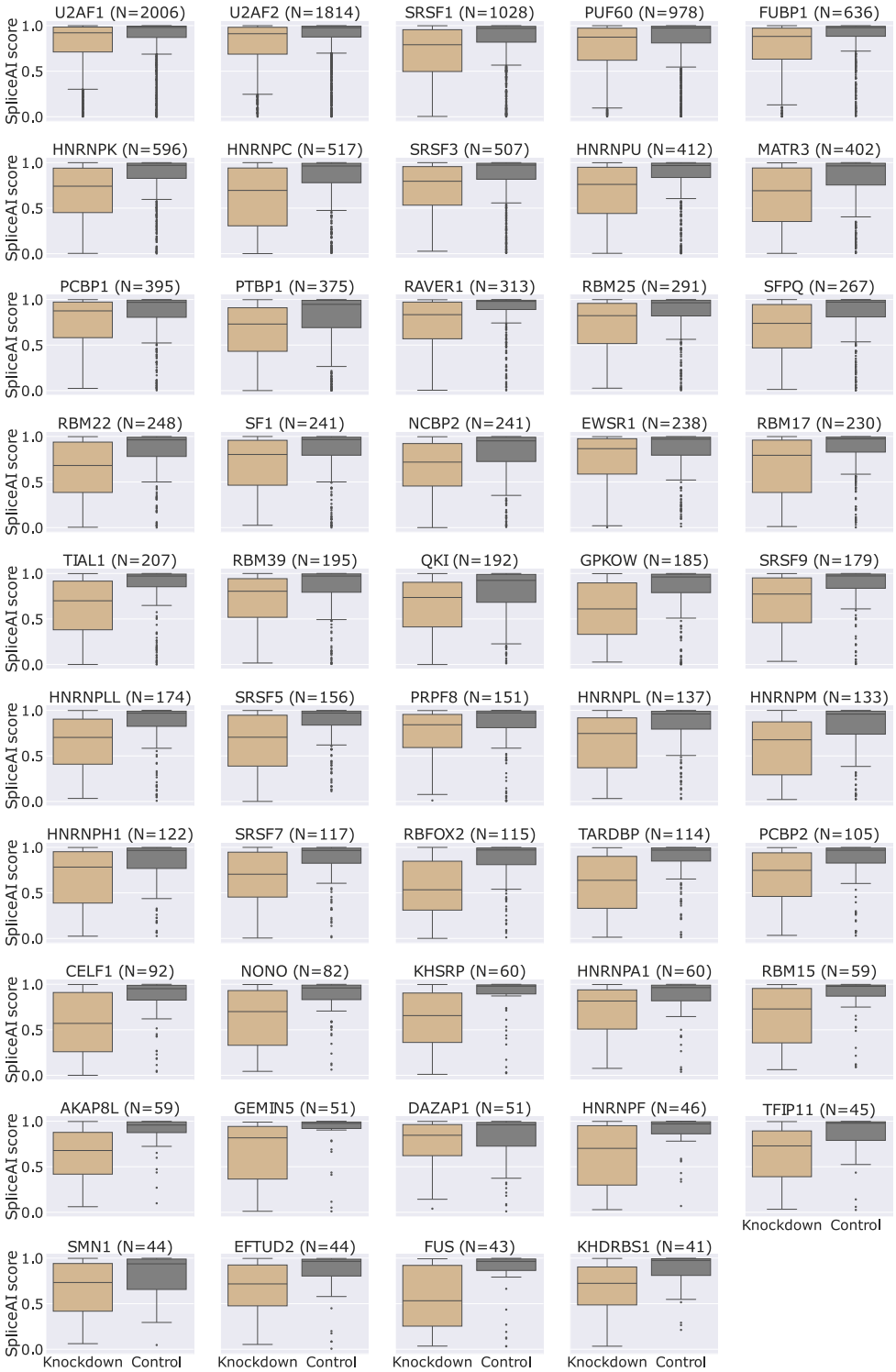

Figure B.6: Distribution of SpliceAI predictions for each exon group across 49 paired datasets. Each datapoint represents a single exon, scored as the mean between the splice acceptor and donor positions. The `N=` at each facet indicates the number of knockdown-sensitive exons. The full dataset size is twice that number.

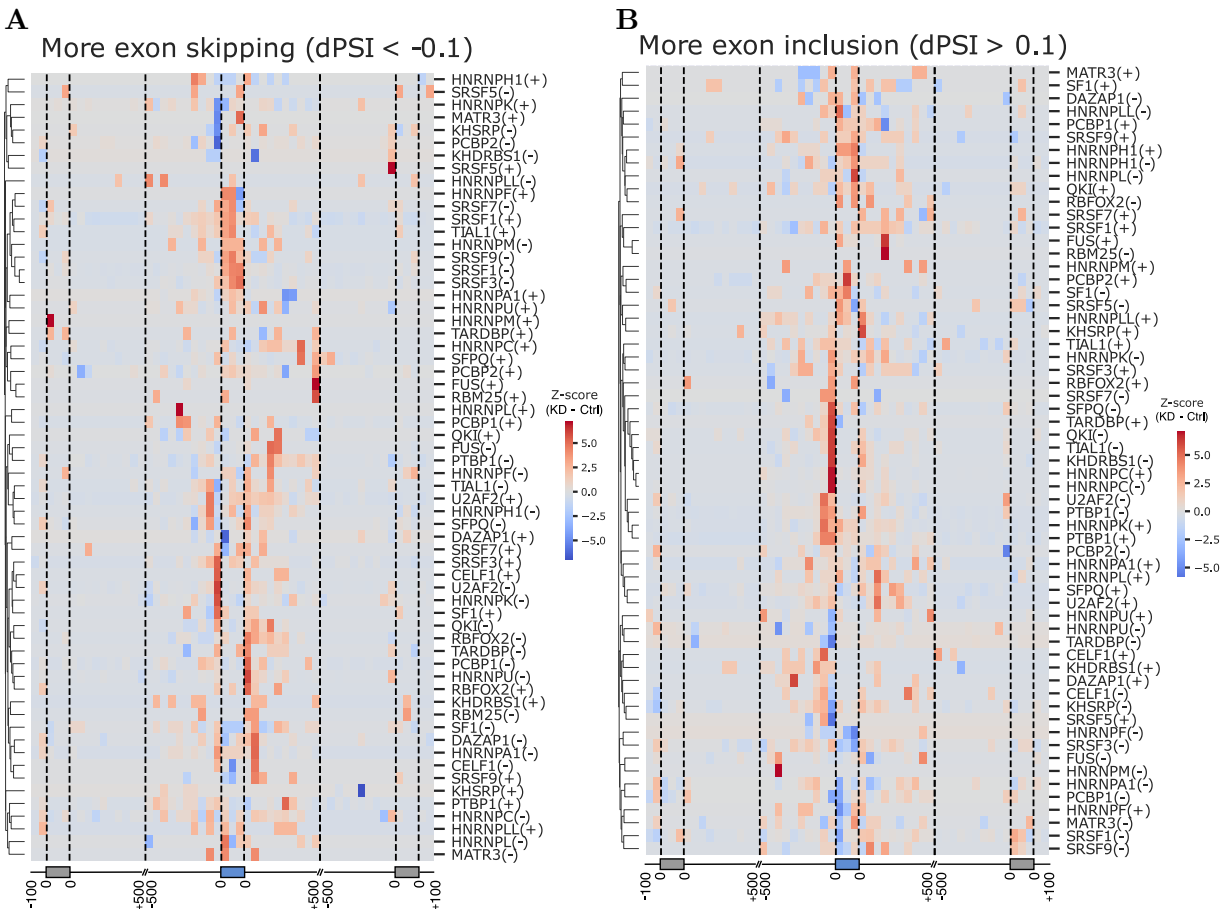

Figure B.7: Position-dependent enrichment of impactful perturbations in knockdown-sensitive vs control sequences for all paired datasets. Positive ($>$ 0.05) and negative ($<$ -0.05) SpliceAI changes were analyzed separately ((+) and (-) strings added to RBP names, respectively). Distance to the upstream/cassette/downstream exons was discretized into region bins of 50bp and the difference in the counts of impactful perturbations between sequences of the knockdown-sensitive and control groups was calculated, for each bin. Heatmap values indicate how much the differences in a particular region deviate from the mean of differences of all regions, for each RBP (row-wise). **A** - Sequences with negative dPSI (more exon skipping)measured in the RNA-Seq data. **B** - Sequences with positive dPSI (more exon inclusion) measured in the RNA-Seq data.

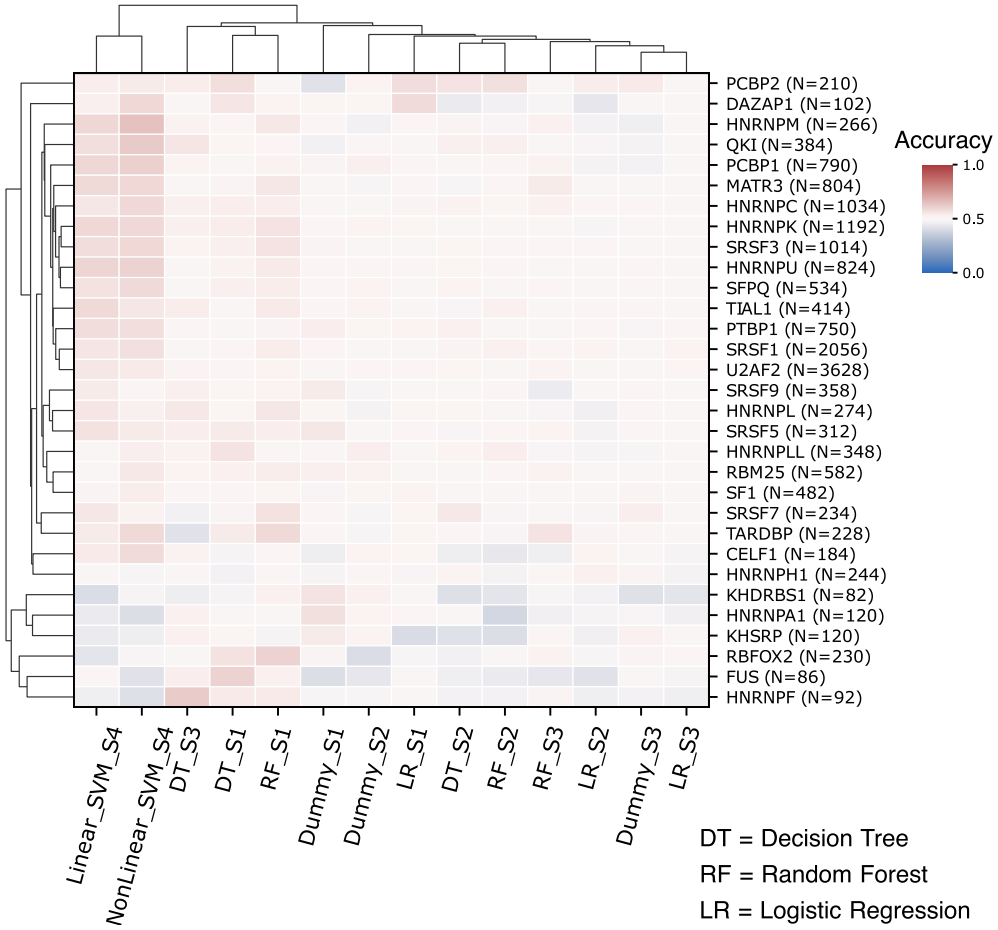

Figure B.8: Tabular machine learning classification to predict exon groups. Like in the regression setting, four different strategies were used to generate sequence-based features. Performance was assessed by computing the mean accuracy of test sets from a stratified cross-validation procedure with 5 splits. Several models were used: a Logistic Regression (with `max_iter=100`), a Decision Tree (with `min_samples_leaf=3, max_depth=5`), a Random Forest (`n_estimators=10`), and two gkm-SVM models, one with a linear kernel (`gkmtrain -t2`), and another employing the RDF nonlinear kernel (`gkmtrain -t3`). Analysis includes also a baseline Dummy classifier set to always predict the most frequent class (although here the datasets are balanced). The `N=` at each row label indicates the dataset size which includes the knockdown-sensitive and control sequences.

Table B.1: ENCODE knockdown identifiers used in the analysis.

RBP name	Cell line	KD experiment	Ctrl experiment	KD bams	Ctrl bams
TARDBP	HepG2	ENCSR527QNC	ENCSR264TUE	ENCFF958MMI;ENCFF965UPD	ENCFF958MMI;ENCFF965UPD
CELF1	HepG2	ENCSR695XOD	ENCSR042QTH	ENCFF819ILY;ENCFF309FMB	ENCFF819ILY;ENCFF309FMB
QKI	HepG2	ENCSR330YOU	ENCSR603TCV	ENCFF709LHN;ENCFF613CGT	ENCFF709LHN;ENCFF613CGT
RBM22	HepG2	ENCSR330KHN	ENCSR997HCQ	ENCFF293GBJ;ENCFF996GEV	ENCFF293GBJ;ENCFF996GEV
SUGP2	HepG2	ENCSR837QDN	ENCSR246RRQ	ENCFF143QAT;ENCFF348XBK	ENCFF143QAT;ENCFF348XBK
TRA2A	HepG2	ENCSR030GZQ	ENCSR491FOC	ENCFF055FJV;ENCFF566QZD	ENCFF055FJV;ENCFF566QZD
PRPF6	HepG2	ENCSR529QEZ	ENCSR491FOC	ENCFF055FJV;ENCFF566QZD	ENCFF055FJV;ENCFF566QZD
DDX5	HepG2	ENCSR808FBR	ENCSR776SXA	ENCFF567LXN;ENCFF427USC	ENCFF567LXN;ENCFF427USC
SRSF3	HepG2	ENCSR376FGR	ENCSR264TUE	ENCFF958MMI;ENCFF965UPD	ENCFF958MMI;ENCFF965UPD
SRSF4	HepG2	ENCSR471INA	ENCSR853AOV	ENCFF299XOT;ENCFF957MEX	ENCFF070YEL;ENCFF379EYH
RBM14	HepG2	ENCSR166MWM	ENCSR521WAI	ENCFF198HLU;ENCFF590DAN	ENCFF198HLU;ENCFF590DAN
TIA1	HepG2	ENCSR057GCF	ENCSR603TCV	ENCFF709LHN;ENCFF613CGT	ENCFF709LHN;ENCFF613CGT
NONO	HepG2	ENCSR647NYX	ENCSR856ZRV	ENCFF435FBM;ENCFF343WPT	ENCFF435FBM;ENCFF343WPT
BUD13	HepG2	ENCSR382QKD	ENCSR067GHD	ENCFF032IND;ENCFF304ITC	ENCFF032IND;ENCFF304ITC
FUS	HepG2	ENCSR927JXU	ENCSR491FOC	ENCFF055FJV;ENCFF566QZD	ENCFF055FJV;ENCFF566QZD
PTBP1	HepG2	ENCSR064DXG	ENCSR603TCV	ENCFF709LHN;ENCFF613CGT	ENCFF709LHN;ENCFF613CGT
RBM47	HepG2	ENCSR711ZJQ	ENCSR279HMU	ENCFF931LCX;ENCFF931YGM	ENCFF931LCX;ENCFF931YGM
TIAL1	HepG2	ENCSR450VQO	ENCSR305XWT	ENCFF894JVP;ENCFF857QEU	ENCFF894JVP;ENCFF857QEU
DDX20	HepG2	ENCSR429OUR	ENCSR775ZWO	ENCFF149SLG;ENCFF221LJC	ENCFF149SLG;ENCFF221LJC
STAU1	HepG2	ENCSR124KCF	ENCSR067GHD	ENCFF032IND;ENCFF304ITC	ENCFF032IND;ENCFF304ITC
U2AF2	HepG2	ENCSR426UUG	ENCSR104ABF	ENCFF905ARG;ENCFF482MPL	ENCFF905ARG;ENCFF482MPL
PCBP1	HepG2	ENCSR635FRH	ENCSR603TCV	ENCFF709LHN;ENCFF613CGT	ENCFF709LHN;ENCFF613CGT
AKAP8L	HepG2	ENCSR807ODB	ENCSR538QOG	ENCFF567UAR;ENCFF958YDS	ENCFF567UAR;ENCFF958YDS
RBFOX2	HepG2	ENCSR767LLP	ENCSR104ABF	ENCFF482MPL;ENCFF905ARG	ENCFF482MPL;ENCFF905ARG
TFIP11	HepG2	ENCSR573UBF	ENCSR856ZRV	ENCFF343WPT;ENCFF435FBM	ENCFF343WPT;ENCFF435FBM
HNRNPL	HepG2	ENCSR155BMF	ENCSR042QTH	ENCFF819ILY;ENCFF309FMB	ENCFF819ILY;ENCFF309FMB
HNRNPA2B1	HepG2	ENCSR769GES	ENCSR042QTH	ENCFF819ILY;ENCFF309FMB	ENCFF819ILY;ENCFF309FMB
SRSF1	HepG2	ENCSR094KBY	ENCSR603TCV	ENCFF709LHN;ENCFF613CGT	ENCFF709LHN;ENCFF613CGT
SRSF5	HepG2	ENCSR781YNI	ENCSR042QTH	ENCFF819ILY;ENCFF309FMB	ENCFF819ILY;ENCFF309FMB
FMR1	HepG2	ENCSR905HID	ENCSR042QTH	ENCFF819ILY;ENCFF309FMB	ENCFF819ILY;ENCFF309FMB
SF1	HepG2	ENCSR644AIM	ENCSR104ABF	ENCFF905ARG;ENCFF482MPL	ENCFF905ARG;ENCFF482MPL
U2AF1	HepG2	ENCSR372UWV	ENCSR067GHD	ENCFF032IND;ENCFF304ITC	ENCFF032IND;ENCFF304ITC
CCAR2	HepG2	ENCSR237IWZ	ENCSR674KEK	ENCFF675TNO;ENCFF432OLY	ENCFF675TNO;ENCFF432OLY
EWSR1	HepG2	ENCSR532ZPP	ENCSR538QOG	ENCFF567UAR;ENCFF958YDS	ENCFF567UAR;ENCFF958YDS
RBM25	HepG2	ENCSR610AEI	ENCSR603TCV	ENCFF709LHN;ENCFF613CGT	ENCFF709LHN;ENCFF613CGT
HNRNPH1	HepG2	ENCSR094HEU	ENCSR194SPW	ENCFF555MSU;ENCFF887DEY	ENCFF555MSU;ENCFF887DEY
SND1	HepG2	ENCSR398LZW	ENCSR491FOC	ENCFF055FJV;ENCFF566QZD	ENCFF055FJV;ENCFF566QZD
NCBP2	HepG2	ENCSR030ARO	ENCSR237YZT	ENCFF745CDS;ENCFF335ETY	ENCFF745CDS;ENCFF335ETY
RBM15	HepG2	ENCSR599PXD	ENCSR585KOJ	ENCFF178VBQ;ENCFF761MBM	ENCFF178VBQ;ENCFF761MBM
PPIG	HepG2	ENCSR620HAA	ENCSR246RRQ	ENCFF143QAT;ENCFF348XBK	ENCFF143QAT;ENCFF348XBK
ZRANB2	HepG2	ENCSR081QQH	ENCSR491FOC	ENCFF055FJV;ENCFF566QZD	ENCFF055FJV;ENCFF566QZD
RBM39	HepG2	ENCSR760EGM	ENCSR104ABF	ENCFF905ARG;ENCFF482MPL	ENCFF905ARG;ENCFF482MPL
SRSF9	HepG2	ENCSR597XHH	ENCSR603TCV	ENCFF709LHN;ENCFF613CGT	ENCFF709LHN;ENCFF613CGT
ADAR	HepG2	ENCSR104OLN	ENCSR305XWT	ENCFF894JVP;ENCFF857QEU	ENCFF894JVP;ENCFF857QEU
HNRNPM	HepG2	ENCSR995JMS	ENCSR067GHD	ENCFF032IND;ENCFF304ITC	ENCFF032IND;ENCFF304ITC
EFTUD2	HepG2	ENCSR620OKS	ENCSR689PHN	ENCFF710REB;ENCFF817MUD	ENCFF710REB;ENCFF817MUD
MATR3	HepG2	ENCSR492UFS	ENCSR237YZT	ENCFF745CDS;ENCFF335ETY	ENCFF745CDS;ENCFF335ETY
SMN1	HepG2	ENCSR090UMI	ENCSR305XWT	ENCFF894JVP;ENCFF857QEU	ENCFF894JVP;ENCFF857QEU
PRPF8	HepG2	ENCSR998MZP	ENCSR856ZRV	ENCFF435FBM;ENCFF343WPT	ENCFF435FBM;ENCFF343WPT
HNRNPK	HepG2	ENCSR853ZJS	ENCSR237YZT	ENCFF335ETY;ENCFF745CDS	ENCFF335ETY;ENCFF745CDS
RBM5	HepG2	ENCSR606PVX	ENCSR481WJH	ENCFF230CCM;ENCFF776AHW	ENCFF230CCM;ENCFF776AHW
CDC40	HepG2	ENCSR278NFF	ENCSR481WJH	ENCFF230CCM;ENCFF776AHW	ENCFF230CCM;ENCFF776AHW
HNRNPF	HepG2	ENCSR693MZJ	ENCSR305XWT	ENCFF894JVP;ENCFF857QEU	ENCFF894JVP;ENCFF857QEU
HNRNPLL	HepG2	ENCSR490DYI	ENCSR264TUE	ENCFF958MMI;ENCFF965UPD	ENCFF958MMI;ENCFF965UPD
SRSF7	HepG2	ENCSR017PRS	ENCSR603TCV	ENCFF709LHN;ENCFF613CGT	ENCFF709LHN;ENCFF613CGT
HNRNPD	HepG2	ENCSR660MZN	ENCSR279HMU	ENCFF931LCX;ENCFF931YGM	ENCFF931LCX;ENCFF931YGM
HNRNPU	HepG2	ENCSR308IKH	ENCSR305XWT	ENCFF894JVP;ENCFF857QEU	ENCFF894JVP;ENCFF857QEU
HNRNPC	HepG2	ENCSR052IYH	ENCSR305XWT	ENCFF894JVP;ENCFF857QEU	ENCFF894JVP;ENCFF857QEU
HNRNPA1	HepG2	ENCSR182DAW	ENCSR305XWT	ENCFF894JVP;ENCFF857QEU	ENCFF894JVP;ENCFF857QEU
SFPQ	HepG2	ENCSR782MXN	ENCSR104ABF	ENCFF482MPL;ENCFF905ARG	ENCFF482MPL;ENCFF905ARG
PPP1R8	HepG2	ENCSR592AQT	ENCSR827PII	ENCFF318QBO;ENCFF968KVV	ENCFF318QBO;ENCFF968KVV
PSIP1	HepG2	ENCSR744YVR	ENCSR674KEK	ENCFF675TNO;ENCFF432OLY	ENCFF675TNO;ENCFF432OLY
RAVER1	HepG2	ENCSR576GOW	ENCSR491FOC	ENCFF055FJV;ENCFF566QZD	ENCFF055FJV;ENCFF566QZD
GEMIN5	HepG2	ENCSR771QMJ	ENCSR689PHN	ENCFF710REB;ENCFF817MUD	ENCFF710REB;ENCFF817MUD
KHSRP	HepG2	ENCSR850CKU	ENCSR237YZT	ENCFF745CDS;ENCFF335ETY	ENCFF745CDS;ENCFF335ETY
PUF60	HepG2	ENCSR648BSC	ENCSR042QTH	ENCFF819ILY;ENCFF309FMB	ENCFF819ILY;ENCFF309FMB
PCBP2	HepG2	ENCSR028ITN	ENCSR491FOC	ENCFF055FJV;ENCFF566QZD	ENCFF055FJV;ENCFF566QZD
DAZAP1	HepG2	ENCSR220TBR	ENCSR042QTH	ENCFF819ILY;ENCFF309FMB	ENCFF819ILY;ENCFF309FMB
FUBP1	HepG2	ENCSR736TAW	ENCSR225PRV	ENCFF560WIR;ENCFF085MQH	ENCFF560WIR;ENCFF085MQH
KHDRBS1	HepG2	ENCSR784FTX	ENCSR279HMU	ENCFF931LCX;ENCFF931YGM	ENCFF931LCX;ENCFF931YGM
RBM17	HepG2	ENCSR385TMY	ENCSR104ABF	ENCFF482MPL;ENCFF905ARG	ENCFF482MPL;ENCFF905ARG
GPKOW	HepG2	ENCSR968YWY	ENCSR264TUE	ENCFF958MMI;ENCFF965UPD	ENCFF958MMI;ENCFF965UPD

Appendix C

Semantically-rich synthetic dataset generation with constrained Genetic Programming

C.1 Adding SQUID as an additional baseline

We included SQUID v0.4.0 [34] as an additional baseline for the generalization experiments. Importantly, SQUID's primary goal is not the generation of the dataset per se; instead, it focuses on using the generated dataset to train surrogate models to interpret genomics deep neural networks. At the dataset generation step, SQUID applies random perturbations to the input sequence. However, it differs from our Random Search approach in several aspects. Firstly, the perturbations are applied directly to the input sequence agnostically to our target specifications - generating diversity in the model prediction space - and thus are much faster to run. Secondly, SQUID restricts perturbations to SNVs and does not allow deletions and deletions. Lastly, it does not allow domain-aware constraints to be applied, such as avoiding perturbations in splice site regions. It was precisely due to this last point that we included SQUID in the comparison: to assess whether allowing perturbations at splice site regions (resulting in drastic changes) would enhance the coverage of the model prediction space.

To make SQUID comparable to our evolutionary-based approaches, we customized a SpliceAI predictor such that the predictions returned are the average of the model at the splicing acceptor and donor positions of the cassette exon. Then, for each input sequence we set up a `RandomMutagenesis` object with a `mut_rate` value that yielded an average number of perturbations per sequence similar to the average edit distance of the datasets obtained for the best GGGP configuration, which was 11.3. Then, we generated 5000 sequences (equivalent to the archive capacity of the evolutionary-based approaches) using 5 different seeds. Finally, we converted the generated sequences into a format suitable for comparison and evaluated

the quality of the datasets accordingly. Of note, we did not run the motif analyses for the SQUID-generated datasets because the perturbation space is different from Random Search and GGGP (splice site regions perturbed), rendering the comparison unfair.

Appendix D

Other contributions

Although the research presented in this document represents the contributions of my PhD studies, I have also participated in other projects. In this section, I outline my specific contributions to these works:

- I analyzed WGS data to identify a heterozygous splicing-disrupting pathogenic intronic variant in MYBPC3, which was subsequently confirmed to cosegregate among family members.

  A. Gomes, **P. Barbosa**, A. Coutinho, I. Cruz, M. Carmo-Fonseca, L. Lopes, "Whole-genome DNA sequencing: The key to detecting a sarcomeric mutation in a 'false genotype-negative' family with hypertrophic cardiomyopathy", *Portuguese Journal of Cardiology*, 2020 [303].

- In a Portuguese cohort of 268 newly diagnosed Acute Myeloid Leukemia patients, a targeted gene panel sequencing strategy was employed to investigate the potential clinical value of molecular screening for risk assessment. I conducted pairwise associations analyzes among somatic mutations to identity significant co-occurrences across gene pairs.

  S. Matos, P. Bernardo, S. Esteves, A. Botelho de Sousa, M. Lemos, P. Ribeiro, M. Silva, A. Nunes, J. Lobato, M.J. Frade, M.Gomes da Silva, S. Chacim, J. Mariz, G. Esteves, J. Raposo, A. Espadana, J. Carda, **P. Barbosa**, V. Martins, M. Carmo-Fonseca, J. Desterro, "Screening a Targeted Panel of Genes by Next-Generation Sequencing Improves Risk Stratification in Real World Patients with Acute Myeloid Leukemia", *Cancers*, 2022 [304].

- An improved reference genome of the cork oak, a tree of significant economic importance in Portugal, has been released. This release includes the first sequences of the chloroplast and mitochondrion organelles. I contributed to the initial analysis of long-read sequencing data generated using PacBio technology and its integration with short-read data from Illumina systems.

A. Usié, O. Serra, P. Barros, **P. Barbosa**, C. Leão, T. Capote, T. Almeida, L. Rodrigues, I. Carrasquinho, J. Guimarães, D. Mendonça, F. Nóbrega, C. Egas, I. Chaves, I.A. Abreu, N.J.M. Saibo, L. Marum, M.C. Varela, J. Matos, F. Simões, C.M. Miguel, M.M. Oliveira, C.P. Ricardo, S. Gonçalves, A.M. Ramos, "An improved reference genome and first organelle genomes of *Quercus suber*", *Tree Genetics & Genomes*, 2023 [305].

- Comprehensive analysis of the erosion of X-chromosome inactivation in female human pluripotent stem cells during differentiation. I performed WES analysis across cell lines to identify SNVs that could be used to assess allele-specific expression in the X chromosome using RNA-seq data.

  A.C. Raposo, P. Caldas, M. Arez, J. Jeremias, **P. Barbosa**, R. Sousa-Luís, F. Água, D. Oxley, A. Mupo, M. Eckersley-Maslin, M. Casanova, A.R. Grosso, S. Teixeira da Rocha, "Erosion of X-Chromosome Inactivation in female hiPSCs is heterogeneous and persists during differentiation", *bioRxiv*, 2024 [306].

- Alternative splicing analysis in Dilated Cardiomyopathy patients with genetic and ischemic origins. I performed most of the bioinformatics analysis in the manuscript.

  M. Furtado*, **P. Barbosa***, T. Carvalho, B. Silva, P. Napoleão, L. Zhang, P. Leszek, M. Carmo-Fonseca, Y. Devaux, S. Martins, "Alternative splicing is similarly dysregulated in heart failure patients with dilated and ischemic cardiomyopathy", *in submission*, 2024.