Big data analytics in bioinformatics

Big Data Analytics in Bioinformatics

Big Data Analytics in Bioinformatics

DISSERTATION

To obtain the degree of Doctor at Maastricht University and the University of Southern Denmark,
on the authority of the Rector Magnificus,
Prof. Dr. Rianne M. Letschert
in accordance with the decision of the Board of Deans,
to be defended in public
on Friday 28th of August 2020 at 12:00 hours in Maastricht

by

Alexander Gulliver Bjørnholt Grønning

Supervisors

Prof. Dr. Jan Baumbach (University of Southern Denmark)

Prof. Dr. Harald Schmidt

Co-supervisor

Prof. Dr. Richard Röttger (University of Southern Denmark)

Assessment Committee

Prof. Dr. Theo de Kok (Chair)

Prof. Dr. Jos Kleinjans

Dr. Markus List (Technical University of Munich, Germany)

Dr. Kedar Nath Natarajan (University of Southern Denmark)

Dr. Josch Paerling (Technical University of Munich, Germany)

Prof. Dr. Qihua Tan (University of Southern Denmark)

Abstract

Advancements in high-throughput technologies have facilitated cost-efficient large-scale productions of biological data in numerous labs worldwide. The production and accumulation of data have led to a big data era – an era where researchers have unprecedented opportunities for learning about biology via bioinformatic analysis of the various and large datasets. But, with the new possibilities comes a range of new challenges that need sophisticated solutions.

Single-cell RNA sequencing (scRNA-seq) data are a quintessential example of a big data type that depends on bioinformatic solutions. Though, many scRNA-seq-specific analytical methods have been invented to process and analyze the data, it has so far been an unexplored endeavor to identify mechanisms that drive the observed single-cell development trajectories. Another challenge that has emerged in the wake of the extensive data productions relates to the thousands of nucleotide variants that have been identified in recent years. The mechanisms by which they affect cellular dynamics are vastly unknown, but important to identify, as they may affect many cellular processes. It is a general bioinformatic challenge to discover patterns that can explain observed biological phenomena. But, with the extensive biological data available it is now possible to synergistically combine different data types and use these to propose new approaches that can predict the outcomes of complicated endogenous signaling pathways.

This thesis presents three manuscripts that introduce new bioinformatic methods and approaches, which seek to address the above-raised challenges. All manuscripts make use of techniques and concepts central to "big data analytics in bioinformatics". The first manuscript presents Scellnetor; Single-cell Network Profiler for Extraction of Systems Biology Patterns from scRNA-seq Trajectories, which is a novel clustering tool for scRNA-seq data. Scellnetor is implemented as an interactive webtool that allows researchers to select and compare single-cell development trajectories. We show that Scellnetor is able to find connected gene subnetworks essential for elucidating differences between distinct cellular development courses.

The second manuscript presents DeepCLIP; a convolutional LSTM (long short-term memory) neural network for analysis of binding preferences of RNA-binding proteins (RBPs). We demonstrate that DeepCLIP produces binding predictions and binding profiles that correlate strongly with *in vitro* and *in vivo* experiments and are sensitive to the effects of nucleotide variants on RBP binding affinity.

The third manuscript explores the potential role of cGMP-dependent protein kinase (PKG) as a reducer of damaging reactive oxygen species (ROS) formation post-stroke. After establishing a crosstalk between PKG- and ROS-signaling, we investigate it by developing a new approach to simulate the downstream transcriptional regulations that takes place upon kinase activity. We predict how expression of transcription factors that regulate gene expression of core ROS-forming enzymes is changed as a result of PKG-activity.

In conclusion, this thesis presents bioinformatic work that proposes solutions to the above-outlined challenges that have arisen with the advent of the big data era. The scientific contributions of this thesis include novel biological insights, new bioinformatic methods and freely available tools that can be readily applied to biological data. Additionally, the thesis includes suggestions to how the presented work might be improved by any researcher who wishes to extend it.

Dansk resumé

Fremskridt indenfor high-throughput-teknologier har faciliteret omkostnings-effektiv generering af biologiske data i adskillige laboratorier verden over. Produktionen og akkumuleringen af biologiske data har ført os til en big data æra – en æra, hvor forskere har nye muligheder for at lære om biologi via bioinformatiske analyser af de forskellige og store datasæt. Men med de nye muligheder følger en række a nye udfordringer, der kræver sofistikerede løsninger.

Single-cell RNA sequencing (scRNA-seq) data er klasseeksemplet på en datatype, der er afhængig af bioinformatisk analyse. Selvom mange scRNA-seq-specifikke analysemetoder er blevet opfundet til at processere of analysere dataene, så har det indtil videre været et stort set uudforsket foretagende at finde mekanismer, der driver de observerede single-cell udviklingsbaner. En anden udfordring, der er dukket op i kølvandet på de omfattende dataproduktioner, vedrører de tusinder af nukleotidvarianter, der er blevet identificeret i de senere år. De mekanismer, hvormed de påvirker nedstrøms genprodukter og cellulære pathways, er stort set ukendte, men alligevel vigtige at identificere, da de kan påvirke og ændre mange cellulære processer. Det er en generel bioinformatisk udfordring at opdage mønstre, der kan forklare observerede biologiske fænomener. Men med de omfattende tilgængelige biologiske data er det nu muligt at synergistisk kombinere forskellige datatyper og bruge disse til at foreslå nye tilgange, der kan forudsige resultaterne af komplicerede endogene signalveje.

Denne afhandling præsenterer tre manuskripter, som introducerer nye bioinformatiske metoder og tilgange, der alle søger at tackle de ovennævnte udfordringer. Alle manuskripter gør brug af teknikker og koncepter, der er centrale for ”big data analytics in bioinformatics”. Det første manuskript præsenterer Scellnetor; Single-cell Network Profiler for Extraction of Systems Biology Patterns from scRNA-seq Trajectories, som er et nyt clustering-værktøj til scRNA-seq-data. Scellnetor er implementeret som et interaktivt webtool, der giver forskere mulighed for at vælge og sammenligne single-cell development trajectories. Vi viser at Scellnetor kan finde gen-subnetworks, der er essentielle for at belyse forskelle mellem forskellige cellulære udviklingsforløb.

Det andet manuskript præsenterer DeepCLIP; et convolutional LSTM (long short-term memory) neuralt netværk til analyse af bindingspræferencer for RNA-bindende proteiner (RBP'er). Vi viser at DeepCLIP producerer outputs, der korrelerer stærkt med *in vitro*- og *in vivo*-eksperimenter, og som er sensitive over for virkningerne af nukleotidvarianter på RBP-bindingsaffinitet.

Det tredje manuskript undersøger den potentielle rolle for cGMP-dependent protein kinase (PKG) som en reducer af den af skadelige reaktiv ilt-art (ROS) dannelse post-stroke. Efter at have etableret en crosstalk mellem PKG- og ROS-signalering, undersøger vi den ved at udvikle en ny tilgang til at simulere de downstream transkriptionelle reguleringer, der finder sted ved kinaseaktivitet. Vi forudsiger, hvordan ekspression af transkriptionsfaktorer, der regulerer

genekspression af core-ROS-dannende enzymer, ændres som et resultat af PKG-aktivitet.

Sammenfattende præsenterer denne afhandling bioinformatisk arbejde, der foreslår løsninger på ovennævnte udfordringer, der er opstået med fremkomsten af big data-æraen. Denne afhandlings videnskabelige bidrag inkluderer ny biologisk indsigt, nye bioinformatiske metoder og frit tilgængelige værktøjer, der let kan anvendes på biologiske data. Desuden indeholder afhandlingen forslag til, hvordan det præsenterede arbejde kan forbedres af enhver forsker, der ønsker at udvide det.

Acknowledgements

First of all, I would like to thank Jan Baumbach for making my PhD-journey possible. It has been fun and instructive to work with him and a pleasure to be supervised by a person who sees solutions instead of problems. Also, it was a nice experience to be part of group where there was plenty of room for being inventive and to follow through with one's own ideas.

I would like to thank Richard Röttger for his intelligent supervision and for his ability to understand my ideas and proposals. It has been pleasant and helpful to work with him and I have really learned a lot during my time as his TA.

I would like to thank Harald Schmidt for introducing me to precision medicine and network pharmacology. These approaches to treatment development can really help a lot of people. It has been a joy to work with him and I am forever grateful for his help finding a birthday present for my father :-)

I would also like to send a special thanks to Tobias, Kathrine, Tim and Thomas for giving me feedback on my thesis drafts. I really appreciate it. I owe you guys a nice beer or two.

Additionally, I would like to thank everyone from the Compbio and Exbio groups and the group of Harald Schmidt for making my PhD-time more fun and for being helpful in times when help was needed.

Last but not least, I would like to thank my significant other for making the writing process easier and for listening to my joyful frustrations along the way.

Contents

Abstract i
Dansk resumé iii
Acknowledgements v
Contents
Overview 1
1 Introduction 3
1.1 Motivation 3
1.2 The central dogma of molecular biology 4
1.2.1 RNA splicing 6
1.3 Transcriptomics 9
1.3.1 RNA sequencing 9
1.3.2 Single-cell RNA sequencing 10
1.3.3 Pre-processing of scRNA-seq expression data and challenges 10
1.4 Biological networks 11
1.4.1 Basic graph theoretical terms and mathematical notations 11
1.4.2 Protein-protein interaction networks and challenges 13
1.4.3 Gene regulatory networks and challenges 13
1.5 Deep learning 14
1.5.1 General structure of deep neural networks 15
1.5.2 Flow of information 15
1.5.3 Feedforward layers 16
1.5.4 Convolutional layers 16
1.5.5 Recurrent layers – focus on long short-term memory layers 19
1.5.6 Output layer and classification 23
1.5.7 Learning with deep learning 23
1.6 Hierarchical clustering 26
1.6.1 Distance, similarity and correlation matrices 27
1.6.2 Dendrogram and finding clusters 28
1.6.3 Linkage types for distance optimization 28
1.6.4 Constrained agglomerative hierarchical clustering 31

Contents

1.7 Dimensionality reduction 31
1.7.1 Principal component analysis 32
1.7.2 Uniform Manifold Approximation and Projection 33
1.8 Hypotheses and aims 34
1.9 References 36
2 Manuscript 1: Comparative single-cell trajectory network enrichment identifies mechanistic patterns in hematopoiesis and CD8 T-cell development 46
3 Manuscript 2: DeepCLIP: Predicting the effect of mutations on protein-RNA binding with Deep Learning 65
4 Manuscript 3: Unraveling of the mechanism behind PKG-dependent regulation of NOX4, 5 gene expression for improved ischemic stroke therapy 98
5 Discussion, outlook and conclusion 116
5.1 References 121
6 Supplementary materials for Manuscript 1 123
7 Supplementary materials for Manuscript 2 136
8 Supplementary materials for Manuscript 3 193
9 Valorization 195
10 Appendix 198
10.1 List of Figures 198
10.2 List of publications 200
10.3 List of abbreviations 201

Overview

Introduction

This chapter describes the overall motivation for the thesis and the manuscripts it contains. This is followed by an introduction to biology and theory relevant to the following chapters. The aim of this chapter is to equip the reader with information and concepts necessary for understanding the included manuscripts. Finally, the hypotheses and aims of the manuscripts are stated.

Manuscript 1: Comparative single-cell trajectory network enrichment identifies mechanistic patterns in hematopoiesis and CD8 T-cell development

This manuscript presents a novel clustering algorithm for scRNA-seq data called Scellnetor. It is implemented as an interactive webtool that allows researchers to select and compare single-cell developmental trajectories. Scellnetor uses a constrained agglomerative hierarchical clustering algorithm to find subnetworks of connected genes that are either similarly or differently expressed through pseudotime in the compared trajectories. Scellnetor can find subnetworks that are crucial for explaining differences between cellular development trajectories. This is demonstrated using data from a study on hematopoiesis and data from a study on dysfunctional CD8 T-cell development in chronic infections.

Manuscript 2: DeepCLIP: Predicting the effect of mutations on protein-RNA binding with Deep Learning

This manuscript introduces DeepCLIP; a convolutional LSTM neural network for the analysis of the binding preferences of RNA-binding proteins. The neural network can detect changes of RBP binding affinity upon the introduction of nucleotide variants. The outputs of DeepCLIP correlate strongly with *in vitro* and *in vivo* experiments and they can be used to guide generation of therapies that treat diseases caused by aberrant splicing. DeepCLIP is implemented as a webtool and as a standalone program.

Manuscript 3: Unraveling of the mechanism behind PKG-dependent regulation of NOX4, 5 gene expression for improved ischemic stroke therapy

This manuscript explores the potential role of cGMP-dependent protein kinase (PKG) as a reducer of formation of damaging reactive oxygen species (ROS) post-stroke. A crosstalk between PKG- and ROS-signaling is establish by *in vitro* hypoxia models. To investigate the observed crosstalk, a new approach to simulate the downstream transcriptional regulations that takes place upon kinase activity is invented. The approach identifies transcription factors that are known for regulating gene expression of ROS-forming enzymes in hypoxic conditions and predicts how their gene expression is changed upon PKG-activity.

Discussion, outlook and conclusion

The bioinformatic tools presented in the manuscripts will be discussed and possible improvements will be proposed. Finally, concluding remarks will be stated.

1 Introduction

1.1 Motivation

Advances in high-throughput technologies have led us to a big data era [1]. Large-scale measurements of cellular events such as messenger RNA (mRNA) expression, protein binding to RNA and protein-protein interactions can now efficiently be obtained at low costs by numerous labs around the world. The production and accumulation of large biological data sets provides unprecedented opportunities for extracting meaningful patterns of complex biological systems and obtaining novel biological and biomedical insights [1]. An important example in this context is single-cell RNA sequencing (scRNA-seq) data, as it allows researchers to analyze inter-cellular relationships with at a hitherto unseen high resolution [2].

The advent of big data and improved computational power have promoted the rise of systems biology, a field that seeks to understand organisms and cells in a holistic manner and cellular events, such as pathways, subprocesses and molecular interactions, in terms of mechanisms [3,4]. Biological networks, like protein-protein interaction (PPI) networks or gene regulatory networks, often are applied in systems biology approaches to elucidate cellular mechanisms that can explain phenomena such as cell differentiation and disease [3,4]. Mechanistic disease causes (endophenotypes) have paved the way for precision medicine, where suitable drugs are identified based on the patient's endophenotypes instead of disease symptoms [5,6]. Interestingly, in-depth systems biology approaches using networks and scRNA-seq data has not yet been meaningfully combined, leaving detailed maps of intra-cellular mechanisms underlying the detected cellular relationships unexplored. Though very useful, many biological networks are biased by overlapping research interests and experimental limitations, which compromise the reliability of the interactions in the networks. Nevertheless, when combined with other supporting data types that adds additional dimensions to the mutual information-pool, it is possible to explore new algorithmic avenues for e.g. simulating biological phenomena. Still, networks are only useful up to a certain point, as some data types are more meaningfully analyzed using completely different approaches. With the large amount of high-throughput sequencing data that have accumulated, millions of nucleotide variants and single nucleotide polymorphisms (SNPs) have seen the light of day [7]. To illuminate the mechanisms by which these affect cellular processes it is essential to develop algorithms that can predict how the variants may perturb formation of downstream gene products, such as mRNAs.

With big data comes big responsibilities - how can the information contained in these big data sets be analyzed and understood in an efficient and meaningful way? And how can we use this information to learn about biology and possibly aid the development of treatments for various diseases? To meet such challenges, it is imperative to construct bioinformatic programs and algorithms that can address the different data types appropriately and uncover meaningful biological patterns in an

optimal way. Additionally, it is important for the progress of science that these algorithmic solutions reach scientists who are not necessarily capable of devising such programs themselves.

This thesis presents bioinformatic work, which is a combination of biology, computer science, statistics, and mathematics. The manuscripts presented in this thesis seek to address the issues outlined in the text above through the utilization of machine learning and computational simulation approaches that can extract biologically relevant patterns from biological data. This section, the Introduction, provides information on the theory and concepts underlying the methods presented in the manuscripts. In the section Discussion, outlook and conclusion, the manuscripts are discussed, and possible improvements proposed. This is followed by final concluding remarks.

1.2 The central dogma of molecular biology

The central dogma of molecular biology is a key concept in molecular biology. It seeks to explain the flow of genetic information between deoxyribonucleic acid (DNA), ribonucleic acid (RNA) and protein. It states that information can transfer between DNA and RNA, but once information has been transformed into protein it

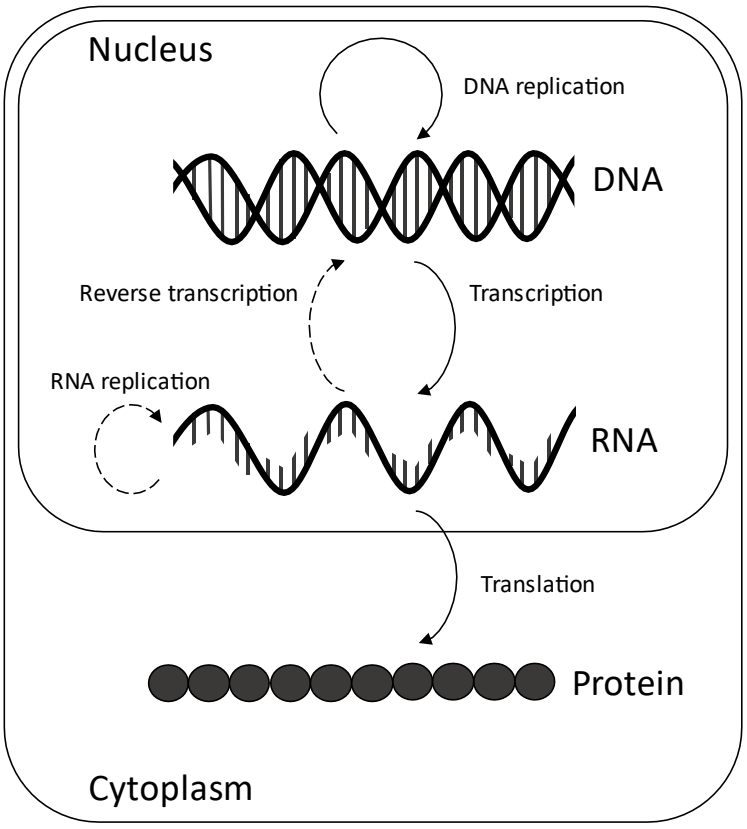


Figure 1. Central dogma of molecular biology in eukaryotic cells. Starting from the top, DNA can replicate itself and be transcribed into RNA. RNA can replicate itself and be translated into protein. In addition, RNA can be reversely transcribed into DNA. DNA and RNA replication and transcription takes place in the nucleus, whereas translation takes place in the cytoplasm. Deoxyribonucleic acid, DNA; ribonucleic acid, RNA.

cannot be transformed back to genetic information [8,9]. The central dogma of molecular biology was proposed by Francis Crick in 1957 and is still used today, as it captures central life-sustaining processes of eukaryotes and prokaryotes [8]. In the following, three important steps of the central dogma (DNA replication, transcription and translation) will be described, starting with a short introduction to DNA and genes. The description of the central dogma is based on [10], unless stated otherwise. The work presented in this thesis focuses on eukaryotic life, so this introduction to the central dogma is based on eukaryotic cells (Figure 1). Genomic DNA consists of two strands of nucleotides. Nucleotides are made up of a phosphate group, a deoxyribose molecule and one of the four nucleobases; adenine (A), thymine (T), guanine (G) or cytosine (C). Typically, DNA has the shape of a double helix, which is the shape that occurs when two connected anti-parallel complementary strands of nucleotides coil around each other. The DNA strands are connected by Watson-Crick base pairings, which means that A binds to T through two hydrogen bonds and C binds to G through three hydrogen bonds. Sequences of specific combinations of nucleobases within genomic DNA constitute genes. A clear-cut universally correct definition of a gene is difficult to find [11]. But for the purpose of providing a clear understanding and to use the term in the context of this thesis, a gene will refer to a DNA sequence that encodes a functional product. Further, gene expression refers to the process where information encoded in the corresponding DNA sequence is used to synthesize a functional product, like a protein or an RNA molecule.

In DNA replication, a single double-stranded DNA molecule produces two identical copies. The double helical structure of DNA is unwound by the enzyme helicase and a replication fork is formed. The synthesis of new strands is catalyzed by the enzyme DNA polymerase, which uses the two disentangled DNA strands as templates for new complementary strands. DNA polymerase moves the replication fork until the “old” strands have been completely separated. The result is two double-stranded DNA molecules, where each double-stranded molecule contains an “old” strand and a newly synthesized one.

The transcription of DNA produces an RNA transcript. The DNA double helix is unwound by helicase and a template strand of DNA is transcribed by the enzyme RNA polymerase. This forms a single-stranded precursor messenger RNA transcript (pre-mRNA). The pre-mRNA molecule is complementary to the transcribed DNA strand with the exception of having a slightly different pentose molecule in its nucleotides (ribose instead of deoxyribose) and the nucleobase uracil instead of thymine. Typically, the pre-mRNA is spliced into a shorter RNA strand, which functions as a recipe for a new protein. RNA splicing will be explained in more depth in section 1.2.1. The RNA splicing forms a new shorter mature mRNA transcript that is transported outside the nucleus where its sequential information is converted into protein. In eukaryotes, transcription of genes is often regulated by transcription factors (TFs). TFs are proteins that bind specific *cis*-regulatory elements on the DNA that typically are located upstream of the gene in a regulatory

region. The TFs are regulators of gene expression, as they can either enhance or suppress it by binding to enhancer elements of silencer elements, respectively.

During the translation process, the mRNA strand binds to a macromolecule called the ribosome. Triplets of nucleobases (codons) are read by transfer RNA molecules (tRNAs) that carry specific amino acids. When the tRNAs read the codons, their attached amino acids are connected like pearls on a string to form a polypeptide. The elongation of the peptide sequence is initiated when it encounters a start codon and ends when a stop codon is encountered. A fully formed protein consists of one correctly folded polypeptide. Multiple proteins may together form larger complexes and enzymes.

Today, it is known that Francis Crick's central dogma of molecular biology cannot account for all flows of genetic information [12]. However, it grants an excellent overview of central cellular processes. This presentation of the central dogma of molecular biology is simplified and does not contain all the dimensions and nuances of DNA replication, transcription and translation. Also, among other regulatory processes of genetic information, reverse transcription and RNA replication have not been explained, as they are beyond the scope of this thesis. Nevertheless, this introduction to the central dogma of molecular biology, will equip readers with the basic information necessary for understanding the work presented in this thesis.

1.2.1 RNA splicing

The formation of functional proteins in eukaryotic cells begins with gene transcription. When genes are transcribed, pre-mRNA sequences are produced. The pre-mRNA molecules are then spliced into smaller mature mRNA sequences that are subsequently translated into proteins. RNA splicing removes introns (non-coding regions) and ligate together exons (coding regions) [13]. This is a dynamic and highly complex process where ordered interactions of numerous factors sequentially assemble and disassemble. These events are carried out by a macromolecular machinery called the spliceosome, which, in the initial splicing-steps, binds to small conserved motifs at the 5' and 3' splice sites (5'ss and 3'ss) located at the exon/intron boundaries, and to the branch site located within the intron [13] (Figure 2). How conserved splice sites are influences how often the adjacent exons are included in the resulting mature mRNA molecule. However, the splice sites are not the only factors that define exons, proper splice sites and regulate splicing [14].

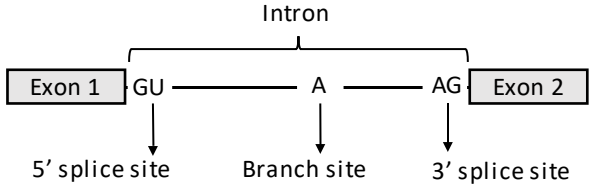


Figure 2. Splice sites and branch point. The figure only shows the most conserved splice site motifs [13].

1.2.1.1 Regulation of RNA splicing

Many factors are involved in deciding which sequence areas are defined as exons and introns [14,15]. These multifactorially based decisions make it possible for a single gene to be spliced into different mRNAs that code for distinctive proteins (alternative splicing) [16], and ensure that essential exons are included constitutively in mature mRNA sequences [15]. Certain *cis*-acting elements, or splicing regulatory elements (SREs), are among the factors that help define exons and introns. Some SREs are enhancer elements and serve as binding sites for proteins that promote exon inclusion. Other SREs are silencer elements, as they induce exon skipping via protein interactions [14,15]. Certain nucleotide variations and mutations can change SREs and disrupt splicing, which may lead to disease [14]. Thus, knowing the binding sites of relevant RNA-binding proteins (RBPs) that bind to SREs can provide information about potential disease causes, which again may lead to the development of a treatment (further elaborated in chapter 3).

1.2.1.2 Binding sites of RNA-binding proteins

Many RBPs recognize more or less conserved sequence motifs in RNA. Sequence motifs are defined as short, recurring patterns in RNA (or DNA) sequences that may represent the binding sites of proteins [17]. The focus of the following is on motifs in RNA sequences. However, the concepts are the same for motifs in DNA sequences. A standard way to represent binding sites of RBPs is via position frequency matrices (PFMs), where each column represents a position of a nucleotide and the rows indicate a nucleotide type. The base letter heights reflect the frequency with which nucleobases occur at that position [17]. Sequence logos are another way to depict binding motifs of RBPs. Here, the letter sizes in the different positions reflect information content. The unit of information is in bits and is a measure of uncertainty [18]. The information at each position of a motif can be found using

$$I_l = 2 + \sum_{b=1}^{4} f_{b,l} \log_2\left(f_{b,l}\right), \tag{1}$$

where $f_{b,l}$ is the frequency of base b at position l converted into a probability. Positions with two bits contain no uncertainty. Figure 3 shows a PFM and a sequence logo that were generated using the same sequences. The nucleobase frequencies captured by PFMs can be used to find new binding sites of RBPs in RNA sequences. This is normally done, by computing a position weight matrix (PWM), where the weights of the nucleobases can be calculated by

$$w(b,l) = \log_2\left(\frac{f_{b,l}}{p_b}\right), \tag{2}$$

where $w(b,l)$ is the weight of nucleobase b at position l, $f_{b,l}$ is the frequency of base b at position l converted in to a probability and p_b is the background probability of base b [17]. PWMs can be used to scan RNA sequences and score potential motifs. The higher the summed score of the PWM, the higher is the predicted binding affinity of the RBP whose binding preferences it represents. PFM, sequence logos and PWMs are conceptually easy to understand and are still used today to represent and find motifs in RNA and DNA [19,20]. A downside to this type of motif representation, is that it does not incorporate information about the sequences that surround binding sites, making it difficult to tell whether a PWM-identified motif is in fact a functional binding site or not. Furthermore, some RBPs tend to recognize structure instead of specific RNA patterns [21,22], which is difficult to represent using standard PFMs (and thereby sequence logos and PWMs). Throughout this thesis, binding preferences of RBPs will also be referred to as "motifs", "binding motifs" or "sequence motifs".

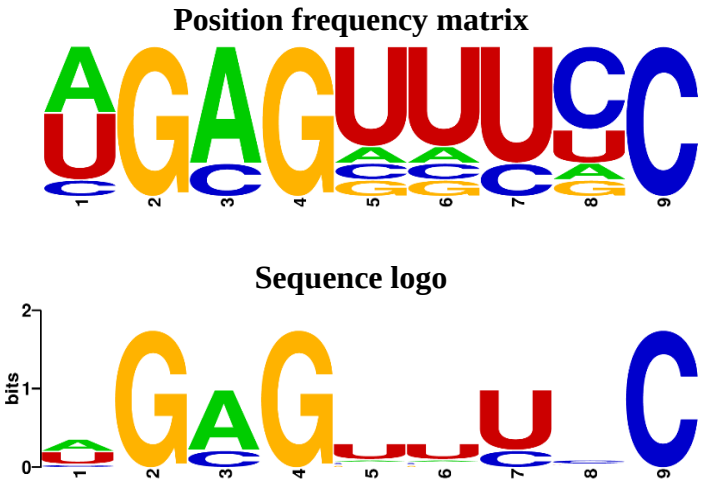


Figure 3. Position frequency matrix and sequence logo[1].

1.2.1.3 Identifying binding sites utilizing CLIP

When investigating the binding preferences of RBPs, one needs to gather sequences to which the RBPs specifically bind. A way to achieve this is via the *in vitro* experiment called SELEX (systematic evolution of ligands by exponential enrichment). In the SELEX procedure, RNA sequences with an affinity for an RBP of interest are iteratively isolated, purified and amplified until only high-affinity RNA sequences are left [23]. However, RBPs cooperate and compete when binding to RNA [15], mechanisms and events difficult to replicate in *in vitro* settings. Thus, studying binding preferences of RBPs *in vivo*, may provide a higher number of sequences that contain functional binding sites. When identifying endogenous RNA sequences that bind to RBPs, procedures that involve crosslinking and

1 Generated using WebLogo [26]

immunoprecipitation (CLIP) are the gold-standard [24]. Many CLIP-procedures exist and most of them include the following steps [24, 25]:

1. UV-irradiation of cells inducing cross-linking of proteins to their binding sites on RNA molecules.
2. Cells are lysed, and the RBP-RNA complexes are immunoprecipitated and purified.
3. Adapters are ligated to the 3' end of the RNA sequences to preclude self-circularization.
4. The RBPs are partially digested by a proteinase K.
5. RNA sequences are converted into cDNA and amplified using polymerase chain reaction (PCR).
6. Amplified cDNA sequences are identified via high-throughput sequencing.

The UV-irradiation captures RBP-RNA complexes as they occur naturally in cells and immunoprecipitation makes it possible to specifically purify the RBPs of interest. High-throughput sequencing allows for the identification of numerous binding sites throughout the genome, often tens of thousands of individual sites. To read more about high-throughput sequencing see section 1.3.1.

1.3 Transcriptomics

The transcriptome can be defined as the complete set of RNA molecules in a cell [27]. The study of transcriptomes (transcriptomics) allows for the monitoring of gene expression as e.g. cells differentiate or upon conditional changes that affect cell homeostasis [27]. For example, a comparison of transcriptional profiles from "diseased cells" with transcriptional profiles from "healthy cells" can provide an estimate of how the investigated disease may affect cells. With analyses like these, researchers can infer gene and protein functions and detect how entire pathways are changed or perturbed by diseases [27]. Typically, differences in gene expression is identified via differential expression analyses, where expression levels of genes in the samples of interest are compared using appropriate statistical tests.

In some studies, the transcriptome is used as proxy for the proteome [28] - the entire set of proteins expressed by cells. In such studies, protein-protein interaction networks may be applied to identify cellular mechanisms based on changes in mRNA expression (exemplified in chapter 2). Even though many mRNA molecules are transformed into functional proteins, they do not mirror actual cellular protein abundance. This is due to alternative splicing, post-translational modifications and differences in the molecular stability of mRNA and proteins, respectively [28].

1.3.1 RNA sequencing

Previously, microarrays were the standard approach for analyzing transcriptomes [27]. Microarrays are sets of defined sequences arranged on a solid substrate that

capture RNA (primarily mRNA) from cells [27]. Today, high-throughput RNA sequencing (RNA-seq) has replaced microarrays, as it allows the sequencing of the entire transcriptome (including small interfering RNAs and long non-coding RNAs) [27,29]. The RNA-seq procedure starts with the extraction of RNA from a relevant sample. The RNA is then copied into more stable double-stranded copy DNAs (ds-cDNAs), which are used for the construction of a fragment library. The sequences in the library are amplified and sequenced (using the desired sequencing technique). The resulting sequences (reads) are then mapped to a reference genome. The number of reads aligned to different genomic areas reflects the expression level of the gene sequence [30]. Compared to microarrays, RNA-seq provides a more detailed analysis of the transcriptome and requires much less starting material [31]. The latter has made single-cell transcriptome analyses possible, which, compared to bulk RNA-seq, can help unravel cellular processes at a much higher resolution [32].

1.3.2 Single-cell RNA sequencing

In conventional bulk RNA-seq studies, the measured RNA levels reflect the average gene expression of the cell ensemble used [33]. However, increasing evidence suggests that gene expression is heterogenous in similar cell types [33–36]. Thus, accounting for cell-to-cell transcriptome variations is important in cellular development studies. With the invention of scRNA-seq, it is now possible to detect transcriptomic differences at single-cell resolutions [33]. The scRNA-seq protocol typically starts by isolating individual cells, which can be achieved through the utilization of different techniques. A commonly used technique is fluorescence-activated cell sorting (FACS), which isolates highly purified single-cells by tagging cells with fluorescent marker proteins [37]. Other techniques apply microfluidics, where nanoliter-sized volumes are used to sort the individual cells [38]. After the single-cells have been lysed and RNA extracted, the RNA is mapped to a reference genome of choice via high-throughput sequencing techniques similar to the ones used in bulk RNA-seq pipelines [32,33].

1.3.3 Pre-processing of scRNA-seq expression data and challenges

Analysis of single-cell expression data can be used to compute cellular developments trajectories and find highly variable genes that drive cell heterogeneity across cells in populations [39] (further elaborated in chapter 2). However, the data are sparser and noisier than expression data derived from bulk RNA-seq [40,41]. Due to the low quantity of RNA in single-cells, it is more difficult to get reliable conversions of transcripts into cDNA, which increases drop-out events where transcripts of genes remain uncaptured [39]. In addition, low quality cells such as cell doublets, broken or empty cells show transcriptome-wide noise compared to high quality cells [42]. Broken cells, for example, have low gene count

and high fractions of mitochondrial counts. Doublets, on the other hand, show unexpectedly high gene counts and large numbers of detected genes [42]. Therefore, single-cell count matrices are subject to certain pre-processing steps such as the filtering of cells with too low or too high gene counts and the removal of cells with elevated expression of mitochondrial DNA. Furthermore, scaling the expression data to unit variance and shifting the mean to zero is often done to prevent the dominance of highly expressed genes in downstream analyses [2, 33,42].

Even though scRNA-seq is noisy, it has led to new innovative algorithms and novel findings in various biomedical fields. Noteworthy examples of novel algorithmic approaches made possible by scRNA-seq include pseudotime calculation and differentiation trajectory inference [43,44]. With these, it has become possible to directly infer inter-cellular relationships based on the individual transcriptional profiles of cell.

1.4 Biological networks

Around 80% of all proteins in the cell interact with other molecules to be fully functional [45]. Many cellular processes such as transcription, replication and splicing are achieved by protein interactions, and alterations in the interactivity of proteins have been shown to cause a range of diseases [46]. Thus, in order to identify the underlying mechanisms of cellular processes, it is vital to study molecular interactions.

Molecular interactions are often studied via biological networks. PPI networks are a type of biological networks that represent PPIs by nodes and edges, where nodes correspond to proteins and edges to interactions. Other types of biological networks include gene regulatory networks (GRNs), whose interactions indicate how different transcription factors might affect the transcription of target genes [47]. Some biological networks, like KEGG [48] or WikiPathways [49], are used to show signaling pathways in more detail, where both proteins and metabolites are included [50]. Only PPI networks and GRNs will be presented in more detail, as they are the only types of biological networks with relevance to the manuscripts presented in this thesis. As indicated above, PPIs and GRNs are essential to the maintenance of cell homeostasis, as they are involved in various cellular processes. Unfortunately, the complete set of PPIs in cells, the interactome, is far from known [51] and causal TF-target gene relationships are difficult to identify [47]. Nevertheless, much scientific work has successfully applied PPI networks and GRNs to e.g. analyze disease associated pathways and find promising drug targets [47,52].

1.4.1 Basic graph theoretical terms and mathematical notations

PPI networks are typically modeled as simple graphs (Figure 4). This means that the edges in networks are unweighted, do not show directionality and cannot be parallel

(two edges between two nodes). Simple graphs do not allow loops (an edge from a node to itself) [53]; however, these are sometimes included in PPI network modelling [54]. GRNs on the other hand, are typically represented as directed graphs that allow parallel edges, as it is important to know which transcription factor regulates which gene [47]. Some biological networks apply edge weighting to convey additional biological information e.g. about interaction confidence or protein co-expression tendencies [55]. The edges are weighted by assigning a numerical value to the individual connections. See Figure 4 for examples of graph types.

Mathematically, a PPI network (undirected network) can be written as follows: if $G = (V, E)$ is a PPI network then the set V contains the vertices (nodes or proteins) and the set E contains the edges, formed by pairs of vertices (interactions) [53].

If $G = (V, E)$ is a GRN (directed network) then then the set V contains the vertices (nodes or proteins or target genes) and the set E contains ordered pairs of vertices (directed edges) [53].

In much research that utilizes biological networks, scientists aim to find subnetworks or subgraphs with relevance for investigated biological conditions. A graph $G_1 = (V_1, E_1)$ is a subgraph of $G = (V, E)$ if $V_1 \subseteq V$ and every edge of $G_1$is also an edge of G. A subgraph, $G_2 = (V_2, E_2)$, is an induced subgraph when $V_2 \subseteq V$ and the edge set, E_2, consists of every edge between the vertices in V_2 [53]. Often it is of interest to find a connected subgraph (see chapter 2). A connected subgraph of a graph is a graph where a path exists between every node pair. A path is a sequence of nodes and edges that connects node A to node B without visiting the same node more than once. For directed graphs, the paths are directed, meaning that one have to follow the directionality of the edges when going from node A to node B [53].

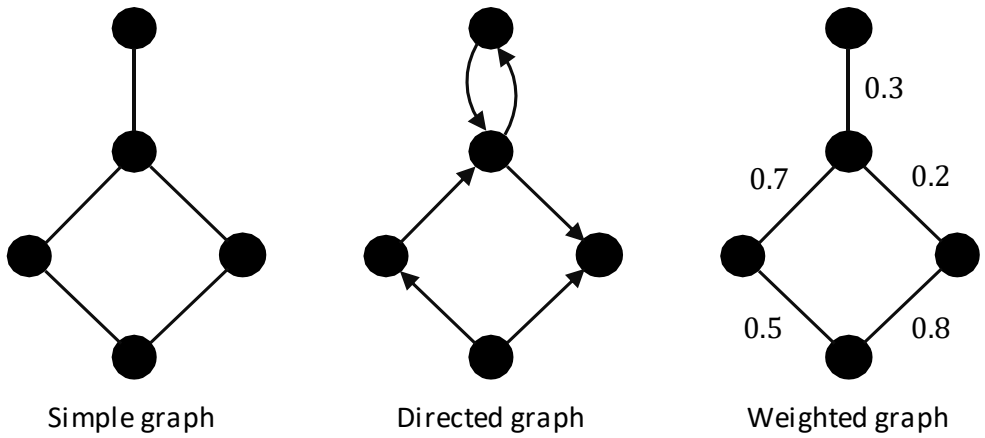


Figure 4. Illustrations of different graph types.

It is said that the physiological importance of proteins can be estimated by the degree number (number of edges) they have in PPI networks [56]. Unfortunately, the connectivity of nodes in such networks may be influenced by different biases that negatively influence the reliability of the interactions.

1.4.2 Protein-protein interaction networks and challenges

In general, biological networks can suffer from two types of biases; technical biases and study biases (also called research bias) [56]. Technical biases are caused by the experimental procedures that detect PPIs. The high-throughput method Yeast two-hybrid (Y2H) screening, for example, detects PPIs in yeast cells (*in vivo*), but it is inclined to detect interactions of proteins located in the nucleus [57]. Another high-throughput method is tandem affinity purification followed by mass spectrometry (TAP/MS), which can detect interactions of protein pairs as well as larger protein complexes. However, the method is prone to miss weaker transient interactions and is inclined to detect interactions between highly abundant proteins [58]. Unfortunately, only a small number of interactions are supported by more than one method type, which adds an additional layer of uncertainty to the collection of PPIs found utilizing these the different methods. Even PPIs sets identified using the same methods have produced poor overlap [45]. Methods that can validate PPIs, like co-immunoprecipitation experiments, are used to filter away false positives. However, these techniques are challenged in other ways that may compromise interaction reliability, and typically they do not detect interactions in a high-throughput manner [45] (making it a tedious endeavor to map the interactome). *In silico* predictions of PPIs can be used as a way to avoid dealing with possible experimentally introduced biases. Even though several features are used in the computation of the interaction predictions (e.g. protein domains, post-translational modifications and Gene Ontology annotations), they still produce many false positives [52,59].

Study biases are caused by overlapping interests of researchers. Some proteins are studied more than others, as they e.g. have shown higher biomedical relevance [56]. Additionally, it has been shown that node degrees of proteins in PPI networks correlate positively with the number of publications that investigate them [60]. Thus, many proteins included in several scientific papers may have a degree that is higher than the average. It is unclear whether the degrees of these “hub nodes” reflect a natural high connectivity, or if this phenomenon is a consequence of the study bias [56]. Some publicly available PPI network databases, like BioGRID [61] and STRING [55], use curation as a way to isolate “true” PPIs.

In general, the interactome-representations from databases like BioGRID and STRING may not exist in such completeness as the networks otherwise indicate [46]. Many methods that detect PPIs do not take subcellular localization into account [45,52], meaning that some PPIs might never occur even though they are physiochemically possible. In addition, expression levels of proteins vary across tissue types, so interactomes of different cells might not be the same [46].

1.4.3 Gene regulatory networks and challenges

GRNs are topological maps that show relationships between TFs and their target genes [62]. It is not a trivial task infer relationships between TFs and gene targets

and thereby to infer a GRN. Thus, many different and sophisticated methods that rely on different data types have been developed to meet the challenge [47]. Some approaches seek to detect co-expressions of genes based on their transcript expression profiles derived from e.g. DNA microarray studies of RNA-seq [63,64]. The co-expression tendencies can be established based on linear Pearson correlations between transcriptional expression data [65], or by assuming non-linear dependencies and use Spearman Correlation [63]. A downside to these GRN inference approaches is that causality is far from established between correlated TFs and genes [66].

Other methods utilize DNA sequence motifs to infer TF-target gene relationships [47]. TF binding site motifs can be represented by PWMs, which are used to score sites on DNA (often within the regulatory regions of genes) and thereby predict binding site positions of TFs [67]. However, the presence of TF binding sites is only slightly predictive of gene expression, as genes' expression levels also rely on additional factors, like other TFs [47]. To this end, the presence of TF binding sites in a regulatory region of a gene does not prove that the TF and the gene in fact interact. Some genes under specific conditions or in certain tissue types are not open for transcription due to epigenetic regulations [47]. But, via usage of Chromatin Immunoprecipitation sequencing (ChIP-seq) or ChIP DNA microarrays (ChIP-chip), maps of functional genome-wide binding sites of TFs of interest can be identified, which can be used to infer GRNs [68].

All inferred GRNs require experimental validations, such as specific TF knockouts or overexpressions or alterations in regulatory elements, before the molecular basis of inferred regulatory edges can be defined [47]. Still, GRNs have shown to be extremely useful for obtaining a systematic understanding of molecular mechanisms in health and disease and to identify new potential therapeutic targets [47, 69] (further elaborated in chapter 3).

1.5 Deep learning

Deep learning and deep neural networks are names for machine learning algorithms that allow computers to build complex concepts out of simpler ones and learn from experience [70]. They can successfully uncover structures in complex and high dimensional data and are used in various fields from medical science to agriculture [71]. Deep neural networks are inspired by biological neural networks (hence the name), however, the two do not have much in common. To avoid any confusion, "neurons" of deep neural networks will be called nodes. The "deep" in deep learning refers to the fact that these algorithms typically have deep network graphs. Depth in this context represents the length of the longest path from input to output [70,72,73]. In the following, deep neural networks will also be referred to as neural networks. In this section, basic deep learning techniques and functionalities will be presented which relate to the work presented in chapter 3. The focus will be on binary classifiers trained using supervised learning techniques.

1.5.1 General structure of deep neural networks

Basic deep neural networks have an input layer, hidden layers and an output layer (Figure 5). The input layer does not contain any "real" nodes or "processing units", but merely functions as a symbolic layer that passes on input data to connected layers. The hidden layers are named this way because their data processing is "hidden" or not visible [70]. However, many have successfully extracted outputs of hidden layers and used these to gain new insights into their data and neural network model (see chapter 3 or [74]). The output layer outputs predictions - the probability that a given input example belongs to a certain class [70,75,76].

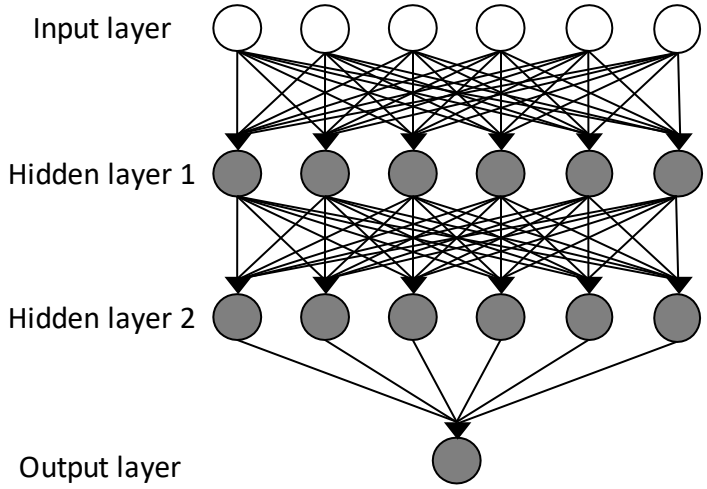


Figure 5. Feedforward neural network graph. All nodes in the network are illustrated as circles. Arrows show the flow of information. At the top, the input layer is shown. The circles (the nodes) are not filled with color, which indicates that these are not "processing units" of the network. The input layer feeds input data to Hidden layer 1. All nodes of Hidden layer 1 receive all input data simultaneously. The same is true for Hidden layer 2, but here the layer receives input from Hidden layer 1. Hidden layer 2 passes its outputs to the Output layer.

1.5.2 Flow of information

The flow of information for the typical deep neural network used for binary classification tasks goes from the input layer to the output layer through hidden layers. This is also referred to as the forward pass of the network [75]. In Figure 5, the first layer from the top is an input layer. It passes the input data to the first connected hidden layer (Hidden layer 1). Each node of the first hidden layer processes the input data and outputs novel representations of the data (new concepts). The new concepts or data representations generated in the Hidden layer 1 are fed to the nodes of the consecutive hidden layer (Hidden layer 2), which build more complex representations of the input data. The "deepest concepts" or most abstract data representations are processed in the output layer, which computes a prediction of the data. In Figure 5, the output layer consists of a single node (more

about this in section 1.5.6). Note that the deep neural network in Figure 5 is in fact not that "deep".

1.5.3 Feedforward layers

Feedforward layers are a type of layers where the flow of information is unidirectional and goes *forward* – from input directly to output [77]. Feedforward layers are also known as "fully connected layers", as all nodes in the layers receive all inputs from the previous layer at the same time (see Figure 5 for feedforward neural network). Other layer types exist that have recurrent connections. These are called recurrent layers and will be described in more detail in section 1.5.5 [77]. Given a single input example from a data set, the output vector, $\boldsymbol{h} = (h_1,\ h_1, h_1 \ldots h_m)$, of a feedforward layer is found by the following vector-matrix product and summation

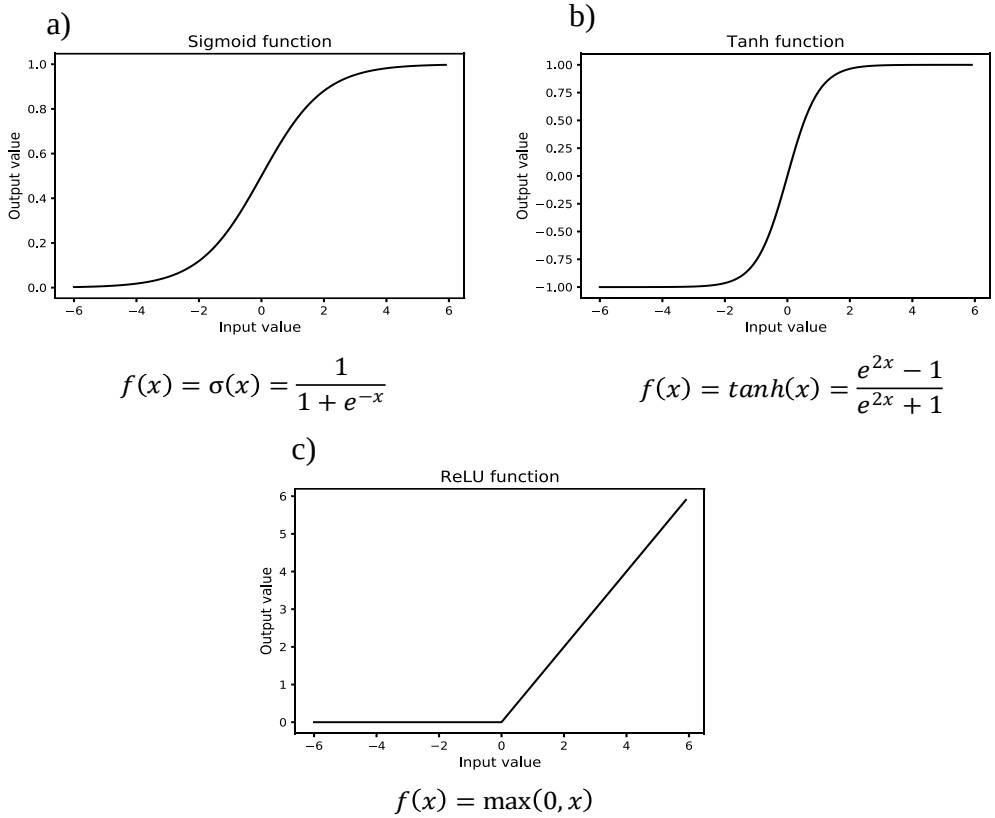
$$\boldsymbol{h} = f(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}), \tag{3}$$

where $\boldsymbol{x} = (x_1, x_2, x_3, \ldots\ x_n)$ is the input vector, $\boldsymbol{W}$ is the weight matrix of the layer with size $m \times n$, $\boldsymbol{b} = (b_1, b_2, b_3, \ldots\ b_m)$ is the bias vector and $f(x)$ is the activation function. $\boldsymbol{W}$ and $\boldsymbol{b}$ are the parameters of the feedforward layer and are the objects that are changed during training to reduce the error produced by the network [76,77] (more about this in section 1.5.7). The number of nodes m is a hyperparameter and should be initialized before the network is compiled. The weight matrices of Hidden layer 1 and Hidden layer 2 in Figure 5 are 6×6 matrices, as there are six nodes and six elements in the input vectors. The activation function $f(x)$ can modify the result of $\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$ in different ways. Activation functions introduce a nonlinearity to the networks, which makes them more powerful classification tools [75,78,79]. The types of activation functions for the different layers in the network are chosen before the neural network is compiled. Figure 6 shows three regularly used activation functions and how they behave given different input values. In the following, all inputs to and outputs of presented neural network layers will be in vector form unless stated otherwise, as this might ease the reader's understanding of the math behind the computation of the layer outputs.

1.5.4 Convolutional layers

Convolutional layers are a type of feedforward layer [80]. But unlike the nodes of feedforward layers presented above, convolutional nodes are not necessarily fully connected. The neural connectivity of cats' visual cortex inspired the invention of convolutional nodes. In certain areas of the visual cortices of cats, neurons are only sensitive to a finite visual field called a receptive field [81,82]. In the same way, convolutional nodes are normally also sensitive to smaller segments of the presented input examples. This is called sparse connectivity. Their visual fields are called

kernels or filters (both are alternative terms for the weight matrices of convolutional nodes) [81–83]. Convolutional nodes can stride along presented input examples and thereby scan the data instances for important features. Figure 7 shows an example



$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$

$$f(x) = tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

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

Figure 6. Three regularly used activation functions. a Sigmoidal activation function. All inputs to the function are squashed into values between 0 and 1. **b** Hyperbolic tangent (tanh) activation function. All inputs are squashed into values between -1 and 1. **c** Rectified linear unit (ReLU) activation function. All negative input values to the function are converted to zeros and all positive input values remain unchanged.

of how a 1D convolutional layer with one node convolve over an input vector. The output value of a convolutional operation can be obtained by

$$z_n = \sum_{i=0}^{|\boldsymbol{w}|-1} x_{n+i} w_{i+1}, \tag{4}$$

where z_n is the output value when the convolution starts at position n in the input vector, $|\boldsymbol{w}|$ is the length of the kernel $\boldsymbol{w}$, x_{n+i} is the input vector element at position $n + i$ and w_{i+1} is the kernel element at position $i + 1$. The output vector of a convolutional layers is also called a feature map [83]. While 2D and 3D convolutional layers exist and are often used in tasks such as image recognition, they are irrelevant for the work in this thesis and will be disregarded. Convolutional layers do not perform real convolutions, where the kernels are flipped before computing the dot products. Instead, cross-correlations are performed. Cross-correlations performed by convolutional layers will, however, be referred to as

convolutions. Further, the two different mathematical operations will result in identical output vectors when used in neural networks.

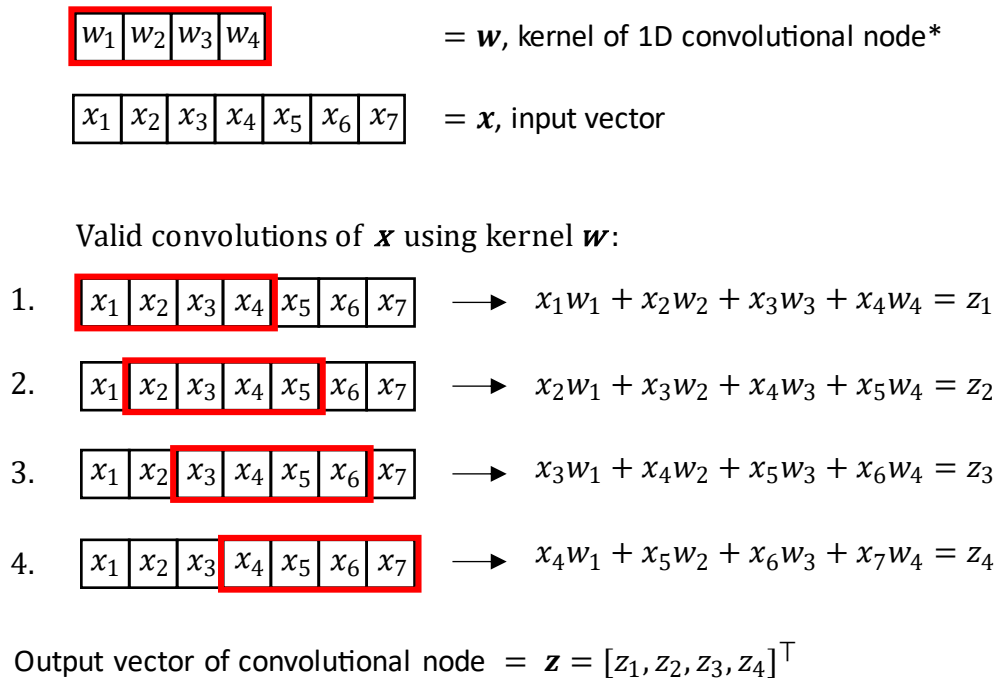


Figure 7. Valid convolutions of the input vector $\boldsymbol{x}$ using kernel $\boldsymbol{w}$. Input vector $\boldsymbol{x}$ and kernel $\boldsymbol{w}$ are shown at the top. Below, the convolutional operations are shown as well as the computation of the output. At the bottom, the output vector $\boldsymbol{z}$ is shown. * = row vector of size 1×4.

1.5.4.1 Convolutional layers are high performing feature detectors

Convolutional layers have three basic qualities that make them high-performing feature detectors: sparse connectivity, parameter sharing and equivariance to translation [83]. Sparse connectivity has already been explained earlier in section 1.5.4. Parameter sharing refers to the usage of the same parameters more than once when computing an output vector (also called "feature map"). For example, note that the same weights (w_1, w_2, w_3, w_4) are used for all the convolutional operations in Figure 7. Equivariance to translation describes the ability to recognize features of the input data independently of their spatial locations in the different data examples. For example, if the input vector $\boldsymbol{x}$ in Figure 7 represents an RNA sequence with 7 nucleotides and the kernel $\boldsymbol{w}$ recognizes the base pattern "UAGG" as a feature of the data. Then the output value of $\boldsymbol{w}$ would be the same regardless of where the motif "UAGG" is located in the input vector. Altogether, the presented qualities of convolutional nodes make them experts at detecting features in input data and at detecting the locations of the features in the data [83].

1.5.4.2 Convolution types – valid, same and full

The convolution type applied to the input vector in Figure 7 is called "valid". Valid convolutional operations use only the data available in the input examples. The output vector sizes of convolutional layers that perform valid convolutions are $|\boldsymbol{x}| - |\boldsymbol{w}| + 1$, where $\boldsymbol{x}$ is the input vector and $\boldsymbol{w}$ is the kernel of the convolutional nodes [83]. This means, the size of the output vector of a convolutional node that performs valid convolutions is smaller than the input vector (unless $|\boldsymbol{w}| = 1$). The use of valid convolutions may limit the number of convolutional layers that can be included in a neural network, as convolutional operations might not be possible once the size of the data flowing through the network has decremented sufficiently [83]. To address this phenomenon, a convolution type called "same" has been invented [83]. With same convolutions, input vectors are zero padded such that the lengths of output vectors are identical to the lengths of the unpadded input vectors. However, the border elements when using valid and same convolutions are in fewer convolutional operations than the vector elements near the middle. "Full" convolutions can be applied to ensure that border elements are not underrepresented in the model. With full convolutions, input vectors are zero padded $|\boldsymbol{w}| - 1$ times on either side. This will produce output vectors of size $|\boldsymbol{x}| + |\boldsymbol{w}| - 1$ [83].

1.5.4.3 Pooling layers

Feature-pooling or pooling are deep learning operations that are often used to isolate certain features detected by convolutional layers [83]. A standard pooling layer type is a max-pooling layer [84] that requires a pooling layer window size as a hyperparameter. The window of the layer strides along a given input vector and extracts the highest value that is within its receptive field. These types of pooling layers might introduce an invariance to translation, as the exact locations of the different convolution-detected features are mixed [83]. To retain some spatial orientation, other pooling layers can be applied. An example could be the winner-takes-all pooling layer, where all values are zeroed out except for the highest one [85]. In general, pooling helps to ensure that the most important features and patterns of the input data are highlighted in the network [83].

1.5.5 Recurrent layers – focus on long short-term memory layers

Recurrent layers are different from feedforward and convolutional layers. They pass output values to successive connected layers and simultaneously send the same outputs back to themselves. Thus, the flow of information is no longer only going forward, but also going backward [75]. Moreover, the nodes from recurrent layers are also inter-connected. Figure 8 depicts a small recurrent neural network (RNN). Different types of recurrent layers exist, but the focus of this subsection will be on the type called long short-term memory (LSTM) layer, since it is the only type that

is of relevance to this thesis. However, a proper introduction to LSTM layers starts by presenting the "vanilla" or standard recurrent layer (will be referred to as "recurrent layer") [75]. Recurrent layers are masters of sequence analysis, as they can "remember" what has been presented already and are thereby able to find context-relevant sequence segments [75]. Inputs to recurrent layers are normally divided into timesteps. An input could be a sequence of T input vectors - a matrix with T rows where every row is an input vector presented at a different timestep. The output vector of a recurrent layer at timestep t, which is in the range $\{t \in \mathbb{N} \mid 1 \leq t \leq T\}$, is obtained by

$$\boldsymbol{h}_t = f(\boldsymbol{W}_x \boldsymbol{x}_t + \boldsymbol{W}_h \boldsymbol{h}_{t-1} + \boldsymbol{b}) \quad (5)$$

where $\boldsymbol{h}_t$ is the output vector, $f(x)$ is the activation function, $\boldsymbol{W}_x$ is the input weight matrix that processes the current input, $\boldsymbol{x}_t$, and $\boldsymbol{W}_h$ is the hidden weight matrix that processes the hidden output vector from the previous time step, $\boldsymbol{h}_{t-1}$, and $\boldsymbol{b}$ is the bias vector [75].

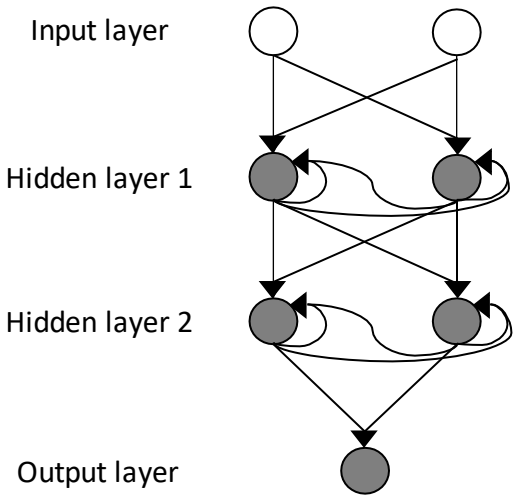


Figure 8. Recurrent neural network. All nodes in the network are illustrated as circles. Arrows show the flow of information. At the top, the input layer is shown. The circles (the nodes) are not filled with color, which indicate that these are not "processing units" of the network. The input layer feeds input data to Hidden layer 1, which consists of recurrent nodes. All nodes of Hidden layer 1 receive all input data simultaneously. All nodes of Hidden layer 1 have cyclic connections and are connected to all nodes in the same layer. The same is true for Hidden layer 2, but here the layer receives input from Hidden layer 1. Hidden layer 2 passes its outputs to the Output layer.

For example, if an RNA-sequence with T nucleotides is analyzed using a recurrent layer, every nucleobase-representation can be introduced at an individual timestep. The complete output matrix (also referred to as output sequence), will be a matrix where every row is an $\boldsymbol{h}_t$ vector. Every $\boldsymbol{h}_t$ vector will have information about every nucleotide in the range $[0; t]$, but with special emphasis on the t^{th} nucleotide [86].

1.5.5.1 Bidirectional layers

When analyzing data sequences, it is often beneficial to remember what has been presented already and to know what lies ahead. For example, in the endeavor of detecting binding sites of RBPs in RNA sequences, it is important to find only the functional sites. To do this efficiently, knowledge of the nucleotide composition of the entire sequence is relevant, since it can help locate contextual clues that only surround functional binding sites. Such sequence information can be obtained by utilization of a bidirectional recurrent layer [75,87]. Basically, a bidirectional recurrent layer consists of two separate recurrent layers that process the same input sequence in a bidirectional manner and that are connected to the same output layer [75]. The forward layer of the bidirectional layer processes the input from the first sequence item to the last, whereas the backward layer processes the input from the last sequence item to the first. Every $\boldsymbol{h}_t$ outputted from a bidirectional recurrent layer will have information about all items in the input sequence, but with special emphasis on the t^{th} item [86].

1.5.5.2 Vanishing gradient problem

Recurrent layers can in theory memorize arbitrarily long input sequences. However, when recurrent layers analyze long sequences, the so-called vanishing gradient problem may occur [75,88,89]. The phenomenon arises because the nodes of the standard recurrent layer are "forced" to remember new inputs whenever they are presented. In practice this means that over time the influence of a given input example on the recurrent layer either decays or blows up exponentially [75]. As a way to circumvent this problem, a new type of recurrent layer was invented where the nodes can "shut off" incoming inputs and even forget what they have learned. This layer type is called an LSTM layer [90].

1.5.5.3 Long short-term memory layers

LSTM nodes are more complicated than the nodes of standard recurrent layers (Figure 9). The nodes of LSTM layers are actually called "memory blocks" [75], but they will be referred to as nodes in this thesis. Where standard RNN nodes have one hidden weight matrix and one input weight matrix, LSTM nodes have four different hidden weight matrices and four different input matrices, and an internal state called cell state. The output vector at timestep t, $\boldsymbol{h}_t$, of a LSTM layer can be computed by

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{xi}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \quad (6)$$

$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_{xf}\boldsymbol{x}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f\right) \quad (7)$$

$$\boldsymbol{g}_t = tanh\left(\boldsymbol{W}_{xg}\boldsymbol{x}_t + \boldsymbol{W}_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g\right) \quad (8)$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t \quad (9)$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{xo}\boldsymbol{x}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \quad (10)$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot tanh\,(\boldsymbol{c}_t), \quad (11)$$

where $\boldsymbol{i}_t$ is the output from the input gate at timestep t, $\boldsymbol{x}_t$ is the input at timestep t, $\boldsymbol{h}_{t-1}$ is the layer output at timestep $t-1$, $\boldsymbol{W}_{xi}$ is the input weight matrix at the input gate, $\boldsymbol{W}_{hi}$ is the hidden weight matrix at the input gate and $\boldsymbol{b}_i$ is the bias vector of the input gate [74]. The same logic can be applied to all the remaining gates, where f is the forget gate, g is the modulatory gate, o is the output gate and c is the cell state. $\sigma(x)$ is the sigmoid function, $tanh(x)$ is the hyperbolic tangent function (Figure 6) and $\odot$ is the Hadamard product (element-wise multiplication).

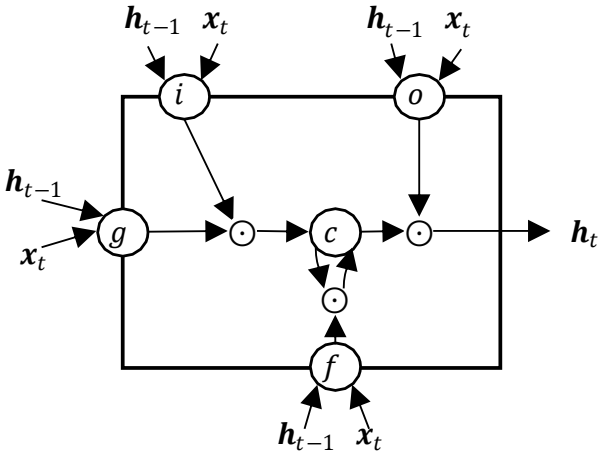


Figure 9. LSTM memory block. $\boldsymbol{i}$ is the input gate, $\boldsymbol{g}$ is the modulatory gate, $\boldsymbol{c}$ is the cell state, $\boldsymbol{f}$ is the forget gate, $\boldsymbol{o}$ is the output gate, $\boldsymbol{x_t}$ is the input at timestep $\boldsymbol{t}$, $\boldsymbol{h_t}$ is the output at timestep $\boldsymbol{t}$, $\boldsymbol{h_{t-1}}$ is the output at timestep $\boldsymbol{t-1}$ and $\odot$ indicates the Hadamard product (element-wise multiplication). The Figure is inspired by Figure 2 in [74].

1.5.5.4 Understanding the LSTM layer - overcoming the vanishing gradient problem

As mentioned previously, by using LSTM layers it is possible to overcome the vanishing gradient problem. This subsection provides an intuitive description of how LSTM nodes might do this and explains the meaning of the term "long short-term memory" [75]. LSTM nodes are not forced to remember every new input or even be influenced by them. For example, if an LSTM node finds that the combination of $\boldsymbol{x}_t$ and $\boldsymbol{h}_{t-1}$ should not help define its cell state, then the output of the input gate (i) will be a value close to zero. Thereby, the input gate will zero-out potential new cell state updates. In continuation of this scenario, if the forget gate (f) remains open (value close to 1), then the node keeps on remembering the already defined cell state (long term memory). On the other hand, if the forget gate outputs a value close to 0, the memorized cell state is wiped clean and can be redefined in

terms of outputs from the input gate and modulatory gate (g). The latter might induce the short-term memory of the LSTM nodes.

1.5.6 Output layer and classification

When applying deep learning to binary classification tasks, the output layer is typically a single sigmoid node. The sigmoid activation function squashes all inputs to values between zero and one, and as the output value is in this range, it can be interpreted as a probability [75,77]. A binary classifier is only able to group input examples into either class 0 (C_0) or class 1 (C_1). The conditional probability that an input example, $\boldsymbol{x}$, is a member of class 1 can be computed by

$$p(C_1|\boldsymbol{x}) = y = \frac{1}{1 + e^{-\boldsymbol{w}\boldsymbol{x}+b}}, \tag{12}$$

where $\boldsymbol{w}$ is the weight row vector of the sigmoid node, $\boldsymbol{x}$ is the input example and b is the bias value of the sigmoid node. The conditional probability that an input example is a member of class 0 can be found by

$$p(C_0|\boldsymbol{x}) = 1 - y. \tag{13}$$

If the probability is ≥ 0.5 the corresponding input example is classified as class 1 and if the probability is < 0.5 the corresponding input example is classified as class 0.

1.5.7 Learning with deep learning

How does a deep neural network learn to predict accurately? In general, deep neural networks need a high number of input examples to gain a good understanding of the data [70]. However, it is difficult to provide a number of examples required to generate a healthy neural network that produces accurate predictions. When training a deep neural network classifier, the applied data are typically divided into three sets; training set, validation set and test set, which is standard procedure when training machine learning models [91]. The sizes of the sets depend on the data and on what is deemed relevant for the classification task at hand.

1.5.7.1 Calculating errors of predictions

When training a neural network, its predictions are held up against labels that indicate the “true” classes of the input examples. The errors of the predictions propagate back through the network and change the weights of its layers and nodes such that future predictions will produce lower errors [76]. An error function is needed to calculate the prediction errors. Different error functions exist, but

common for all of them is that they compare network predictions with data labels and produce nonnegative (error) values [76]. Error functions are also called cost of loss functions [75,76]. In this section, only the error function called binary cross entropy error function will be considered, as it the only one of relevance to this thesis. In addition, it has been found to work well when the output layer is a sigmoid node, as it help the nodes to "learn" faster [92]. Binary cross entropy is given by

$$E(\theta) = -L\ log\ (y)\ -(1-L)\ log(1-y), \tag{14}$$

where θ is all the parameters of the network, L is the target label and y is the probability of the output layer. The aim of the training is to find the global minimum of the error function, $E(\theta)$. $E(\theta)$ depends on the parameters of the neural network, so during training the network iteratively updates the parameters in order to minimize the general error produced by the network [92].

1.5.7.2 Backpropagation and stochastic gradient descent

To effectively change the parameters, an update algorithm is applied. Many such algorithms exist and many of them are based on, or inspired by, the one called stochastic gradient descent (SGD), whose basic principles will be outlined here. As previously stated, the objective of the training is to find the global minimum of $E(\theta)$. With the SGD algorithm, the partial derivatives of $E(\theta)$ with respect to all parameters of the neural network are computed (the gradient of $E(\theta)$). The gradient vector is denoted ∇E. The partial derivatives of $E(\theta)$ with respect to all parameters of the neural network can be found using the chain rule [92], which is a formula that can compute the derivatives of composite functions – like deep neural networks. In fact, a neural network can be seen as one big composite function. For example, the output of the feedforward neural network depicted in Figure 5 can be found by

$$p(C_1|\boldsymbol{x}) = \sigma(\boldsymbol{w}^3\ f^2(\boldsymbol{W}^2 f^1(\boldsymbol{W}^1\boldsymbol{x} + \boldsymbol{b}^1) + \boldsymbol{b}^2) + b^3), \tag{15}$$

where $\boldsymbol{W}^1$ and $\boldsymbol{b}^1$ are the parameters of Hidden layer 1, $\boldsymbol{W}^2$ and $\boldsymbol{b}^2$ are the parameters of Hidden layer 2, $\boldsymbol{w}^3$ and b^3 are the parameters of the output layer, $f(x)^1$ and $f(x)^2$ are the activation functions of Hidden layer 1 and Hidden layer 2, respectively, $\sigma(x)$ is the sigmoid function and $\boldsymbol{x}$ is the input vector. The gradient vector, ∇E, contains information about how the neural network parameters should be changed in order to reduce $E(\theta)$ [76]. See Figure 10 for an example of the mechanisms. The change of the parameters can be obtained by

$$\theta' = \theta - \eta\ \nabla\ E, \tag{16}$$

where θ is the parameters of the network, ∇E is the gradient vector and η is the learning rate. The learning rate is a small positive scalar that determines the size of the parameter changes [93]. Finding an optimal η can be difficult. A typical goal is

to find a η that is small, but not too small as this would make the SGD algorithm work too slowly. If η is too large on the other hand, it may cause the training error to increase [76,94]. The update algorithm is called "stochastic" gradient descent, because the training set is further divided into randomly generated mini-batches. By dividing the input data into mini-batches, neural networks gain impressive speed-ups when training as well as when predicting [76]. It is typically the mean or the sum of the error produced by the items in the mini-batches that propagates back through the networks and changes the parameters of the layers.

The parameters of recurrent layers are updated using an algorithm called backpropagation through time [95]. It is similar to standard backpropagation, as the parameter changes can be computed using the chain rule. But, with recurrent layers, $E(\theta)$ depends on how the nodes in these layers activate themselves through the T timesteps. So, the error with respect to the parameters of a recurrent layer are found by summing the sequence of derivatives that is calculated starting with $t = T$ and ending with $t = 1$ [75].

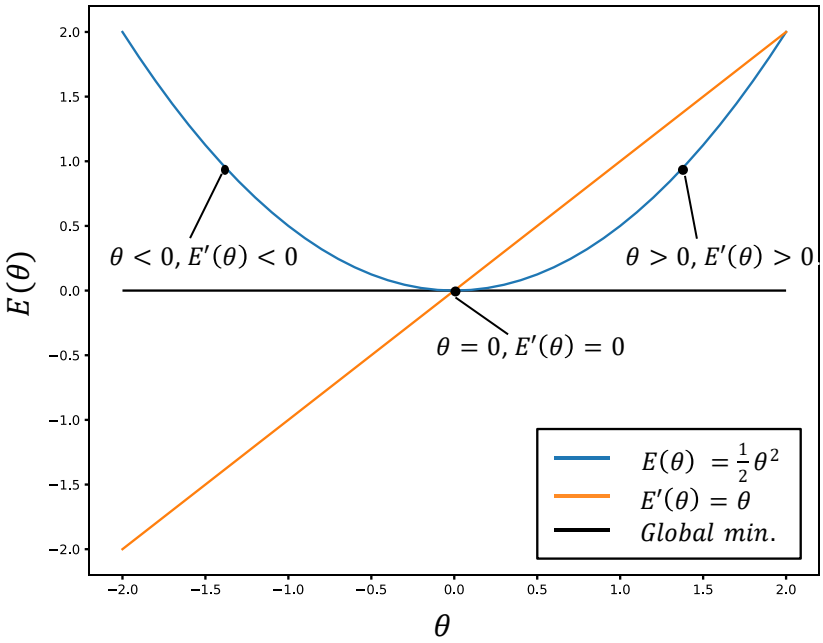


Figure 10. Aim of gradient descent. $\boldsymbol{E(\theta)}$ is an error function that has a global minimum at $(\mathbf{0}, \mathbf{0})$. By taking the derivative of $\boldsymbol{E(\theta)}$ it is possible to identify how $\boldsymbol{\theta}$ should be changed such that the global minimum can be reached. If $\boldsymbol{E'(\theta) > 0}$ then $\boldsymbol{\theta}$ should be decreased, while $\boldsymbol{\theta}$ should be increased if $\boldsymbol{E'(\theta) < 0}$.

1.5.7.3 Generalization, overfitting and regularization

When training a deep neural network, the error decreases as the network learns produce the desired outputs. However, if a model is trained too intensely on the same data, it loses its ability to generalize. This means it can produce a low error on the training data and a much higher on the validation and test set, which represent similar but yet unseen data examples. This phenomenon is called overfitting [91,92]. To continuously check the generalizability of the neural network, a prediction of the items in the validation set is performed after the network has trained on all mini-

batches of the training set. When the neural network has iterated through all data of the training set and validation set, it has run an epoch. Typically, a neural network trains for many epochs before the training is complete. A trick to avoid overfitting is to save the parameters that produce the lowest validation error, thereby saving the model that generalizes the best (early stopping) [96]. As a final test of the trained model, the test set is predicted by the network. In an optimal scenario, the error (or accuracy) of the prediction of the test set is high and is similar to the accuracy of the prediction of the validation set.

Another way to reduce the validation and test error (sometimes at the expense of the training error), is by use of regularization strategies [96]. Common regularization strategies include L^2 and L^1 regularizations, which add the squared L^2 norm (squared Euclidean distance) and the L^1 norm (sum of absolute values), respectively, of chosen layers to the total computed prediction error. The regularization techniques are controlled by a small scalar (a hyperparameter similar to the learning rate) that modifies the magnitude of the regularization value they add to the total error pool. In general, these strategies keep the weight-values of the network's parameters low, while ensuring that the network focuses on important data features [96]. Dropout is another powerful regularization technique that can be applied to the individual layers of a neural network [96,97]. By dropping out (temporarily removing) p percent of random nodes from a neural network layer whenever a new mini-batch is fed to the network during the training phase, the network learns to rely on all nodes in that layer instead of only some of them. This technique can markedly reduce overfitting in various classification tasks and it works well on many layer types [97].

1.6 Hierarchical clustering

The goal of clustering is to discover natural groupings of a data set [98]. Hierarchical clustering is a clustering method that structures a hierarchy of the analyzed data. Hierarchical clustering algorithms are typically either divisive or agglomerative [99]. With divisive algorithms, all data items are initially assigned to one big cluster. The cluster is then iteratively split into smaller clusters until every item is in its own cluster. Agglomerative clustering algorithms work the other way around; every data item starts in its own cluster and the clusters are iteratively fused together one by one until all clusters have been amalgamated into one cluster. Only agglomerative hierarchical clustering algorithms are relevant for the work presented in this thesis; therefore, divisive hierarchical clustering algorithms will not be described any further.

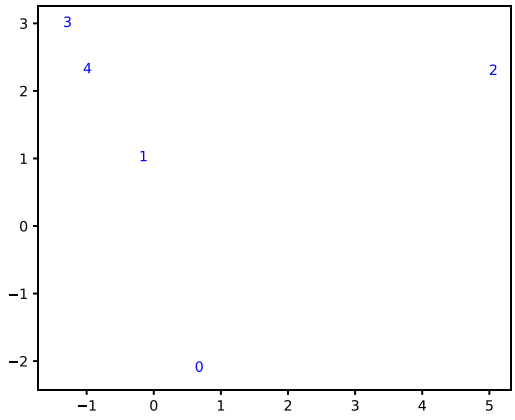
An important quality of hierarchical clustering is that it is deterministic. This means that the same data set will always produce the same hierarchy given that the same hyperparameters are applied. This differs from non-deterministic clustering approaches whose resulting clusterings might vary from run to run even though the same data is analyzed [99,100]. An example of a non-deterministic clustering

algorithm is K-means [98]. Hierarchical clusterings can be conducted in many different ways and in many varieties. In this section, only the approaches relevant to the manuscripts in this thesis will be presented. Data points and data items will be used interchangeably in the following and refer to the individual items in a data set.

1.6.1 Distance, similarity and correlation matrices

Agglomerative hierarchical clustering algorithms base their hierarchy structuring on the data points' distances, similarities or correlations [99]. Whether distance, similarity or correlation is used depends on how the hierarchical clustering algorithm has been constructed. To compute, for example, a distance matrix of n data points, one calculates the distances between all possible pairs of data items, which results in an $n \times n$ matrix [101]. Figure 11 shows data points (0, 1, 2, 3, 4) and a Euclidean distance matrix calculated using the data points. For a metric (distance function), such as Euclidean distance, values lie in the range $[0: \infty[$, where zero is no distance. For a similarity function (e.g. Jaccard index) the values lie in the range $[0: 1]$, where 1 is maximal similarity. And for a measure of correlation (e.g. Pearson's correlation) the values range between $[-1: 1]$, where -1 is anticorrelation and 1 is perfect correlation. When distance metrics are used for clustering, a typical goal is to partition objects in close proximity in the same cluster and to partition objects far away from each other into distinct clusters, such that within-cluster distance is small and the between-cluster distance is large [102]. The same principle holds true for data point similarities and correlation measures. Here, the objective is to find clusterings where the within-cluster similarity or correlation is high/positive and the between-cluster similarity or correlation is low.

a)



b)

	0	1	2	3	4
0	0	3.22	6.20	5.46	4.72
1	3.22	0	5.36	2.28	1.54
2	6.20	5.36	0	6.38	6.04
3	5.46	2.28	6.38	0	0.74
4	4.72	1.54	6.04	0.74	0

Figure 11. Scatter plot and associated distance matrix. a Scatter plot showing position of the point **0**, **1**, **2**, **3** and **4**. **b** Distance matrix based on the points shown in a) calculated using Euclidean distance.

1.6.2 Dendrogram and finding clusters

Typically, the hierarchies created by hierarchical clustering algorithms are visualized using dendrograms. Dendrograms are tree-graphs where the terminal vertices (the data items that are clustered) are called leaves and top edge is called the root [99]. Figure 12 shows examples of dendrograms, where the names of the data items are shown on the x-axes and the distances are shown on the y-axes. The height of the edges that connects vertices or clusters reflects the between-cluster relationships [99] (between-cluster distances in Figure 12). The inter-cluster relationships depend on the linkage type used for the clustering (more on this in section 1.6.3).

A challenge when using the hierarchical clustering method is to decide or calculate the "optimal number of clusters". One way to achieve this, is to manually inspect the dendrogram and choose a place to "cut the tree" and thereby form a number of clusters. Another way is to predefine a number of clusters and their minimum sizes before the clustering starts. In this scenario, the dendrogram-structuring will stop when the user-chosen threshold has been reached (more on this in chapter 2). It is possible to compute a pseudo-optimal cut-off value or number of clusters. Pseudo is used in this context, because it is uncertain whether an optimal solution exists. The endeavor to find a clustering optimum for the hierarchical clustering approach has preoccupied many researchers over the years and many possible solutions exist [99]. However, it is not of relevance to this thesis, so it will not be discussed any further.

1.6.3 Linkage types for distance optimization

When clustering data with hierarchical clustering algorithms, one needs to decide which linkage type to use. A linkage defines how the algorithm agglomeratively fuses together data items and thereby generates dendrograms and clusters. For example, if a hierarchical clustering is conducted based on an $n \times n$ matrix, the algorithm will merge clusters $n - 1$ times before the clustering is complete. Whenever a merge is done, new inter-cluster distances that depend on the linkage type will be calculated [76]. Below, some commonly used linkage types are presented. See Figure 12 for examples of dendrograms calculated using different linkage types.

1.6.3.1 Single linkage

Single linkage is a strategy that always fuses the nearest clusters together [99]. The distance between clusters containing more than one item is equal to the distance between the two nearest items in the clusters (Figure 12a). The single linkage strategy tends to produce relatively large clusters compared to other linkage types (see below). An advantage of single linkage is that it is insensitive to ties, meaning

that equal distances between several points will have the same distance or similarity in the dendrogram.

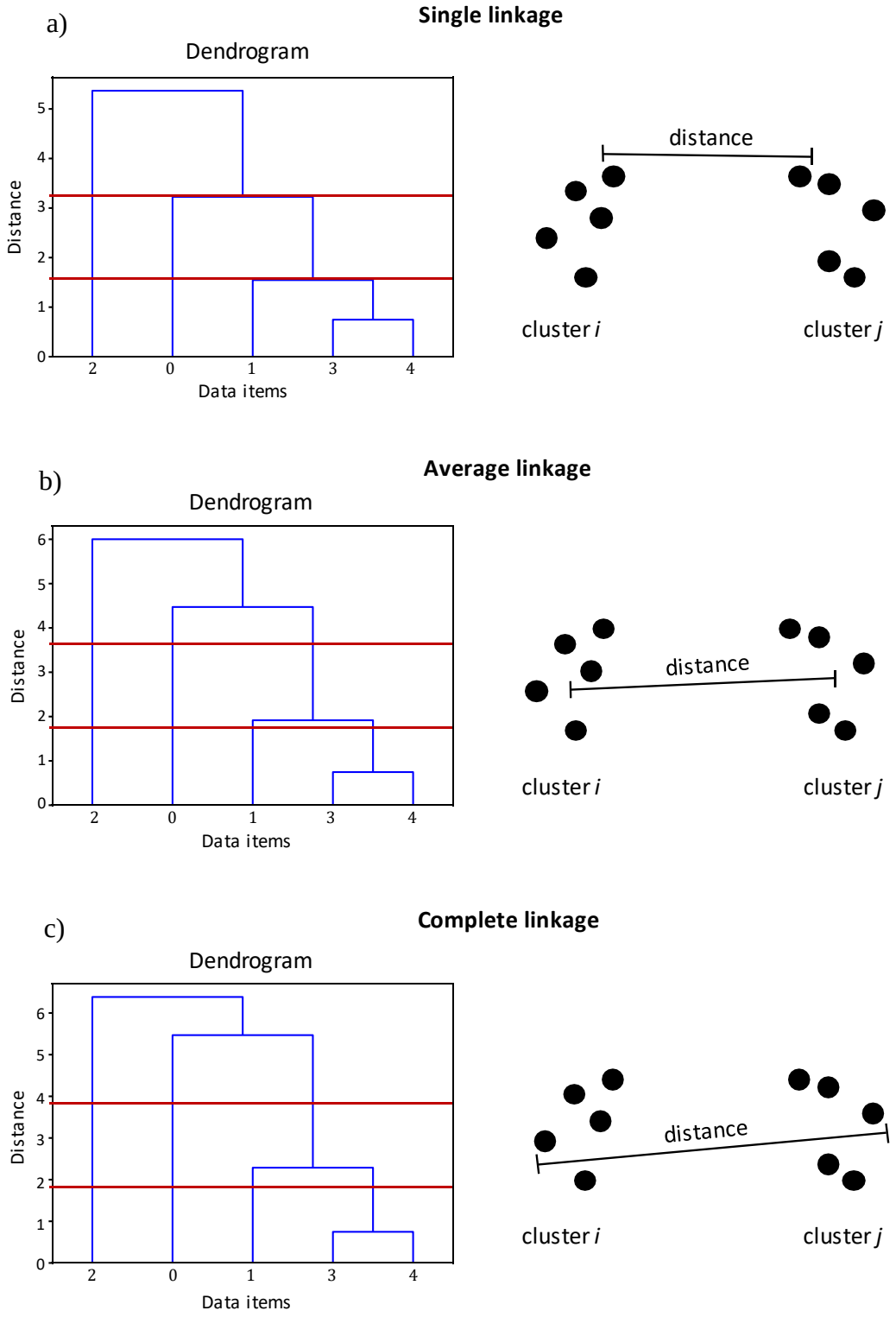


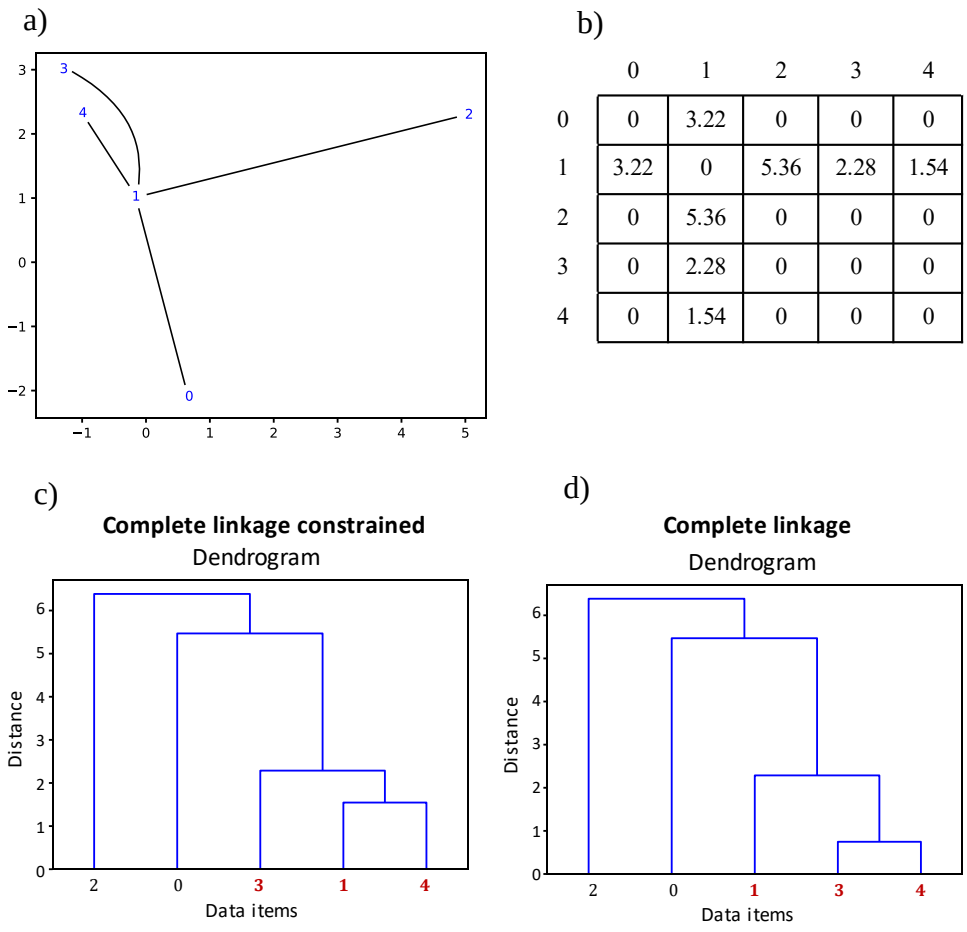
Figure 12. Dendrograms and illustrations of single, average and complete linkage. a-c (left) dendrograms found using simple, average and complete linkage, respectively. The red horizontal bars match merge-distances of the single linkage dendrogram and they make it easier to see the different linkage types' impact on the resulting dendrogram. **a-c** (right) Illustrations of single, average and complete linkage. All dendrograms are calculated using the data points shown in Figure 11a.

1.6.3.2 Complete linkage

With complete linkage [99], the distance between two clusters is equal to the distance between the two cluster objects furthest away from each other (Figure 12c). For all clusters, the method finds the distances of all cluster objects furthest away from each other and then fuses together clusters with the smallest distance. Complete linkage is more prone to producing smaller, denser and more similar-sized clusters.

1.6.3.3 Average linkage

Average linkage is a combination of the two linkage strategies described above [99]. The method finds the average distances between all clusters and fuses together clusters with the smallest distance (Figure 12b). For example, if we have three data points (1, 2 and 3) and 1 and 2 have been fused into a cluster, then the new average-linkage-distance is the average distance of point 1 and 2 to point 3.



	0	1	2	3	4
0	0	3.22	0	0	0
1	3.22	0	5.36	2.28	1.54
2	0	5.36	0	0	0
3	0	2.28	0	0	0
4	0	1.54	0	0	0

Figure 13. Scatter plot with connected data points, constrained distance matrix and complete linkage clustering with and without constraints. a Scatter plot with connected data points. **b** Distance matrix reflecting the distance between initial possible cluster merges. **c** Dendrogram of constrained agglomerative hierarchical clustering given the point in a and using the complete linkage. **d** Dendrogram of agglomerative hierarchical clustering given the point in a without the connections and using the complete linkage.

1.6.4 Constrained agglomerative hierarchical clustering

Constrained agglomerative hierarchical clustering is useful when something is known *a priori* about the connectivity of the data items [99]. For example, assuming that the points 0, 1, 2, 3 and 4 in Figure 13a are genes and that their distances from each other reflect gene expression similarities – the closer they are to each other the more similar are their gene expression patterns. Unlike Figure 11a, Figure 13a shows how the five genes are connected in a hypothetical PPI network. The goal is to cluster the genes using hierarchical clustering where the step-wise cluster merges are constrained by the connectivity of the genes. As with normal agglomerative hierarchical clustering, a distance matrix is needed. Here, however, only possible connections are shown (Figure 13b). After each merge, the distance matrix is updated based on the inter-cluster distances and updated cluster connectivities. At the first cluster merge, genes 1 and 4 are fused into a cluster. This cluster's connectivity and inter-cluster distances will be updated using the connectivity and distances of both genes 1 and 4. This continues until all genes have been amalgamated into one cluster. The resulting dendrogram can be seen in Figure 13c.

1.7 Dimensionality reduction

Many biological data are represented by vectors or matrices in high-dimensional space. For example, an scRNA-seq expression matrix typically has more than 1000 columns and rows corresponding to cells and genes, respectively. High-dimensional data in general introduce certain problems when it comes to visualization and clustering (two important tools for data analysis in bioinformatics). Data visualization is difficult to perceive and comprehend when working with more than three dimensions. Increased dimensionality makes it difficult for clustering techniques to find the data's underlying heterogeneity [103]. Especially, clustering algorithms that apply distance metrics are heavily affected by high dimensionality, since distance differences between data points diminish as dimensionality increases (the curse of dimensionality) [104]. Dimensionality reduction can preserve high data variability in a smaller number of variables by mapping the high-dimensional data onto a lower-dimensional space [103,105]. Dimensionality reduction techniques are essential to much bioinformatic work that seeks to extract meaningful patterns from data in high-dimensional spaces [103,105,106]. However, dimensionality reduction also reduces the information in the data [33]. Therefore, is important to find the optimal dimensionality reduction technique. Different techniques for dimensionality reduction of biological data exist [105–107]. The work in this thesis applies PCA (principal component analysis) and Uniform Manifold Approximation and Projection (UMAP) [107]; therefore only these techniques will be presented in more detail. PCA and UMAP are often used in combination, where the output of PCA is used as an input to UMAP [2,105]. For this reason, in the following presentation of PCA and UMAP, PCA will be introduced before UMAP.

1.7.1 Principal component analysis

PCA has been the most frequently applied dimensionality reduction techniques for many years and is still widely used today [106]. It is a simple non-parametric method based on linear algebra that can reduce the dimensions of high-dimensional data sets. PCA computes the principal components (PCs) of data, which are orthogonal vectors that each capture an amount of variance of the data. The aim of employing PCA for the dimensionality reduction of high-dimensional data is to represent them using a number of PCs smaller than the number of dimensions that the source data have. To make the most informative characterization of the data, the PCs that capture the highest degree of variance should be used [108]. When visualizing data using PCs, the high-dimensional data is typically plotted along the two most variance-capturing PCs.

Principal components can be found in different ways. The following presents an approach that relies on Eigenvector decomposition [108]. We assume an $m \times n$ matrix $\boldsymbol{X}$ that contains biological data, e.g. scRNA-seq data, where m is the number of cells and n is the number of genes. We want to find the two PCs that capture the most variance of the cells' gene expression patterns. When conducting a PCA on $\boldsymbol{X}$, the aim is to find an orthogonal matrix $\boldsymbol{P}$, where the columns of $\boldsymbol{P}$ are the PCs of $\boldsymbol{X}$. First, we want to find the covariance matrix of $\boldsymbol{X}$, which can be obtained by

$$\boldsymbol{C_X} = \frac{1}{n-1}\big((\boldsymbol{X} - \overline{\boldsymbol{x}})^T(\boldsymbol{X} - \overline{\boldsymbol{x}})\big), \tag{17}$$

where $\overline{\boldsymbol{x}} = (\bar{x}_1, \bar{x}_2, \bar{x}_3, \dots \bar{x}_n)$ is the mean vector and $\bar{x}_i$ is the mean of column i in $\boldsymbol{X}$. The covariance matrix is a symmetric $n \times n$ matrix, where the diagonal is the variance of the gene expression across cells of the individual genes. The off-diagonal terms are the variances of the different genes' expression across cells. The next step is to find the Eigenvector and Eigenvalue of the covariance matrix. This can be done via

$$\boldsymbol{C_X v} = \lambda \boldsymbol{v}, \tag{18}$$

where $\boldsymbol{v}$ is the Eigenvector and λ is the Eigenvalue. The Eigenvalue can be found via the following steps:

$$\boldsymbol{C_X v} = \lambda \boldsymbol{v} \tag{19}$$

$$\Leftrightarrow \boldsymbol{C_X v} = \lambda I \boldsymbol{v} \tag{20}$$

$$\Leftrightarrow \boldsymbol{C_X v} - \lambda \boldsymbol{I v} = 0 \tag{21}$$

$$\Leftrightarrow (\boldsymbol{C_X} - \lambda \boldsymbol{I})\boldsymbol{v} = 0, \tag{22}$$

where $\boldsymbol{I}$ is the identity matrix of size $n \times n$. If $\boldsymbol{v}$ is a non-zero vector, λ can be found by solving for λ given $|\boldsymbol{C_X} - \lambda \boldsymbol{I}| = 0$, where $|\boldsymbol{C_X} - \lambda \boldsymbol{I}|$ is the determinant of $\boldsymbol{C_X} - \lambda \boldsymbol{I}$. The vector $\boldsymbol{v}$ can be found by solving for $\boldsymbol{v}$ given $\boldsymbol{C_X v} = \lambda \boldsymbol{v}$. When solving for

λ given $|\boldsymbol{C_X} - \lambda \boldsymbol{I}| = 0$, one needs to solve a polynomial with degree n, which results in an "Eigenvalue-vector" of length n that contains different Eigenvalues (one for every solution of the polynomial). Similarly, once $\boldsymbol{v}$ has been found, the different PCs can be obtained by calculating $\lambda \boldsymbol{v}$ using all possible λ. The result is a $n \times n$ matrix, $\boldsymbol{P}$, where columns corresponds to Eigenvectors and PCs (the principal components are the Eigenvectors of the covariance matrix $\boldsymbol{C_X}$).

It is important to sort the Eigenvalues and Eigenvectors such that the highest Eigenvalues and corresponding Eigenvectors can be found, since these capture the most variance of the analyzed data. In this example, we wanted to find the two largest Eigenvalues and corresponding Eigenvectors. How much of the source data the different PCs explain can be computed by dividing the Eigenvalues by the sum of the "Eigenvalue-vector". This will show how much the different PCs explain in terms of percentage, where 1 is 100%.

The two highest ranking Eigenvectors form a novel matrix, $\boldsymbol{W}$, where the Eigenvectors constitute the columns. Thus, $\boldsymbol{W}$ has the shape $n \times 2$ and can be used to calculate the 2D coordinates of cells based on the two most informative PCs. This can be done via the matrix multiplication

$$\boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}. \tag{23}$$

The coordinates in $\boldsymbol{V}$ can be plotted and clustered.

1.7.2 Uniform Manifold Approximation and Projection

UMAP is a more recent dimensionality reduction method. It has proven superior to many other techniques on various types of data, as it can effectively preserve the local and global data structure in its low-dimensional representations and has a short runtime [105–107]. In contrast to PCA, UMAP is a nonlinear dimensionality reduction technique, which makes it easier to avoid overcrowding of data points in the dimensionality reduced representations [106]. UMAP is based on sophisticated mathematical concepts [107,109] and a thorough explanation of its method is beyond the scope of this thesis; therefore only the basics of the algorithm will be outlined here.

UMAP assumes a uniform distribution of the high-dimensional data points across the manifold[2] that underlies the data, as this allows for easier approximation of their topology[3]. Given *k* nearest neighbors (provided as the hyperparameter, $n\_neighbors$), UMAP computes distances between the high-dimensional data points. UMAP uses the distances to form a weighted graph where an edge is put between every point and its *k* nearest neighbors and where every edge is weighted by the distance it represents. Due to the structure of the high-dimensional manifold and the assumption that the data is uniformly distributed, the distances between the

2 A manifold is a geometric object which locally has a topological structure like Euclidean space [110].
3 Topology refers to the general geometrical properties of spaces [111].

different points are incompatible. This means, the distance from point a to b can differ from the distance from b to a. Some points in the graph are therefore connected by parallel edges that are weighted differently. In high dimensional spaces, the ratio of the distances of the nearest and farthest neighbors to a given data point is almost 1 for a wide variety of data distributions and distance functions (the curse of dimensionality) [104]. UMAP circumvents the curse of dimensionality by focusing on the distance differences of the data points instead of the absolute distances. Therefore, UMAP converts the distances between data points and their k nearest neighbors into values between 0 and 1. The weighted edges of the graph are "re-weighted" such that the value 1 is given to the edge between the data points' closest neighbors, and values < 1 are given to the edges between the data points and their more distant neighbors. The edges that connects to the data points and their neighbors furthest away are given the lowest value and non-edges are given a weight of 0 (their weights are used in later computations). In the next step, the parallel edges are merged to form a single edge that is weighted by the union of their weights. The union is obtained by calculating $a + b - a \cdot b$, where a and b are edge weights of the parallel edges. This weighting of the edges in the graph forms a fuzzy topological representation of the data, which captures the topology of the manifold underlying the high-dimensional data.

In the last steps of the algorithm, the edge weights of the graph are transformed into lower-dimensional distances in Euclidean space, which represents the source data's overall topology. UMAP approach this by turning the transformation into an optimization problem using stochastic gradient descent (see section 1.5.7.2) and cross entropy error function (similarly to equation (14)), where the edge weights are used as probabilities. This means, UMAP compares the edge weights from before and after the unions were found and finds a global state where the total error is a low as possible. In other words, UMAP find low-dimensional distances between data points that are as similar to the high-dimensional distances as possible. For the calculation of the low-dimensional distance between data points, UMAP requires another hyperparameter, min_dist, that controls how close the low-dimensional data points are allowed to be to each other. By iterating through the sets of edges and reducing the error of the error function, UMAP finds a low-dimensional data representation that accurately represent the high dimensional source data.

1.8 Hypotheses and aims

This thesis contains three manuscripts that each presents a novel bioinformatic tool or approach that can analyze biological data and produce new biological and biomedical insights. The following will present some motivating background and descriptions of the main aims and hypotheses that were discussed prior to the implementations of the programs.

Aim 1. scRNA-seq data have enabled cellular developments to be studied at an unprecedented resolution. However, no bioinformatic tools can use the data to find subnetworks of connected genes that explain differences between development trajectories. We aim to provide a systems biology tool that fills this gap and can be used for precision medicine approaches as an extractor of endophenotypes. A goal is to make the tool freely available and easy to use and to provide a platform where researchers can interactively select the trajectories they wish to investigate and compare. We hypothesize that a good solution to the unraveling of subnetworks is to use an agglomerative hierarchical clustering algorithm constrained by the interactions in a PPI network. We further hypothesize that differences in gene expression of selected cell sets can be computed and represented using our hyper-similarity matrix approach, which is based on distances in UMAP space.

Aim 2. Many novel nucleotide variants have been revealed with the advances in high-throughput technologies. Nucleotide variants can affect how proteins bind to RNA and induce change of vital cellular processes, like gene expression. Previously, binding motifs of RBPs were represented using position frequency matrices (PFMs). However, PFMs do not incorporate contextual information and are typically only useful for capturing sequence-specific motifs. We aim to develop a classifier that uses repetitive patterns and sequence contexts to learn the binding preferences of RBPs, and thereby move away from the confinements of the PFM motif representation. Additionally, we aim to construct an algorithm that can be used to predict RBP affinity changes introduced by nucleotide variants in a freely available and easy-to-use platform. We hypothesize that deep learning techniques can help to efficiently solve this task. Also, we aim to construct a "transparent" neural network, where the outputs of the layers can be directly interpreted in the context of the overall classification task. Finally, we aim to construct a classifier whose outputs should agree with experimental data.

Aim 3. It is known that reactive oxygen species (ROS) can regulate signaling of cGMP-dependent protein kinase (PKG) by scavenging of nitric oxide (NO) and damaging the cGMP-forming NO-receptor, soluble guanylate cyclase (sGC). However, as of yet it is unknown whether PKG in turn can regulate ROS formation. Identifying if such a crosstalk exists can help to alleviate post-stroke complications, where ROS overexpression serves as a key patomechanism. Using hypoxia models that simulate post-stroke conditions, we aim to establish whether PKG-activity affects ROS-production. We aim to elucidate this relationship by utilization of biological networks that can help revealing underlying mechanisms that explain any observed inter-regulatory connection. As kinases are known for promoting down- or upregulation of genes, we hypothesize that we can use a GRN to approach this analysis.

1.9 References

1. Greene CS, Tan J, Ung M, Moore JH, Cheng C. Big Data Bioinformatics. Journal of Cellular Physiology. 2014. pp. 1896–1900. doi:10.1002/jcp.24662

2. Luecken MD, Theis FJ. Current best practices in single-cell RNA-seq analysis: a tutorial. Mol Syst Biol. 2019. Available: https://www.embopress.org/doi/abs/10.15252/msb.20188746

3. Altaf-Ul-Amin M, Afendi FM, Kiboi SK, Kanaya S. Systems biology in the context of big data and networks. Biomed Res Int. 2014;2014: 428570. doi:10.1155/2014/428570

4. Likić VA, McConville MJ, Lithgow T, Bacic A. Systems biology: the next frontier for bioinformatics. Adv Bioinformatics. 2010; 268925. doi:10.1155/2010/268925

5. Leopold JA, Maron BA, Loscalzo J. The application of big data to cardiovascular disease: paths to precision medicine. J Clin Invest. 2020;130: 29–38. doi:10.1172/JCI129203

6. Ghiassian SD, Menche J, Chasman DI, Giulianini F, Wang R, Ricchiuto P, et al. Endophenotype Network Models: Common Core of Complex Diseases. Sci Rep. 2016;6: 27414. doi:10.1038/srep27414

7. Zhao Y, Wang K, Wang W-L, Yin T-T, Dong W-Q, Xu C-J. A high-throughput SNP discovery strategy for RNA-seq data. BMC Genomics. 2019;20: 160. doi:10.1186/s12864-019-5533-4

8. Cobb M. 60 years ago, Francis Crick changed the logic of biology. PLoS Biol. 2017;15: e2003243. doi:10.1371/journal.pbio.2003243

9. Crick F. Central dogma of molecular biology. Nature. 1970;227: 561–563. doi:10.1038/227561a0

10. Widłak W. Molecular Biology - Not Only for Bioinformaticians. Springer; 2013. pp. 31–106.

11. Portin P, Wilkins A. The Evolving Definition of the Term “Gene”. Genetics. 2017;205: 1353–1364. doi:10.1534/genetics.116.196956

12. Koonin EV. Does the central dogma still stand? Biol Direct. 2012;7: 27. doi:10.1186/1745-6150-7-27

13. Will CL, Lührmann R. Spliceosome structure and function. Cold Spring Harb Perspect Biol. 2011;3. doi:10.1101/cshperspect.a003707

14. Cartegni L, Chew SL, Krainer AR. Listening to silence and understanding nonsense: exonic mutations that affect splicing. Nat Rev Genet. 2002;3: 285–298. doi:10.1038/nrg775

15. Conti LD, De Conti L, Baralle M, Buratti E. Exon and intron definition in pre-mRNA splicing. Wiley Interdisciplinary Reviews: RNA. 2013. pp. 49–60. doi:10.1002/wrna.1140

16. Roy B, Haupt LM, Griffiths LR. Review: Alternative Splicing (AS) of Genes As An Approach for Generating Protein Complexity. Current Genomics. 2013. pp. 182–194. doi:10.2174/1389202911314030004

17. D'haeseleer P. What are DNA sequence motifs? Nat Biotechnol. 2006;24: 423–425. doi:10.1038/nbt0406-423

18. Schneider TD, Stephens RM. Sequence logos: a new way to display consensus sequences. Nucleic Acids Res. 1990;18: 6097–6100. doi:10.1093/nar/18.20.6097

19. Heinz S, Benner C, Spann N, Bertolino E, Lin YC, Laslo P, et al. Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities. Mol Cell. 2010;38: 576–589. doi:10.1016/j.molcel.2010.05.004

20. Bailey TL, Boden M, Buske FA, Frith M, Grant CE, Clementi L, et al. MEME SUITE: tools for motif discovery and searching. Nucleic Acids Res. 2009;37: W202–8. doi:10.1093/nar/gkp335

21. Hori T, Taguchi Y, Uesugi S, Kurihara Y. The RNA ligands for mouse proline-rich RNA-binding protein (mouse Prrp) contain two consensus sequences in separate loop structure. Nucleic Acids Res. 2005;33: 190–200. doi:10.1093/nar/gki153

22. Buckanovich RJ, Darnell RB. The neuronal RNA binding protein Nova-1 recognizes specific RNA targets in vitro and in vivo. Mol Cell Biol. 1997;17: 3194–3201. doi:10.1128/mcb.17.6.3194

23. Tuerk C, Gold L. Systematic evolution of ligands by exponential enrichment: RNA ligands to bacteriophage T4 DNA polymerase. Science. 1990;249: 505–510. doi:10.1126/science.2200121

24. Ule J, Hwang H-W, Darnell RB. The Future of Cross-Linking and Immunoprecipitation (CLIP). Cold Spring Harb Perspect Biol. 2018;10. doi:10.1101/cshperspect.a032243

25. Huppertz I, Attig J, D'Ambrogio A, Easton LE, Sibley CR, Sugimoto Y, et al. iCLIP: Protein–RNA interactions at nucleotide resolution. Methods. 2014. pp. 274–287. doi:10.1016/j.ymeth.2013.10.011

26. Crooks GE. WebLogo: A Sequence Logo Generator. Genome Research. 2004. pp. 1188–1190. doi:10.1101/gr.849004

27. Blumenberg M. Introductory Chapter: Transcriptome Analysis. In: Blumenberg M, editor. Transcriptome Analysis. IntechOpen; 2019. doi:10.5772/intechopen.85980

28. Payne SH. The utility of protein and mRNA correlation. Trends Biochem Sci. 2015;40: 1–3. doi:10.1016/j.tibs.2014.10.010

29. Bayega A, Fahiminiya S, Oikonomopoulos S, Ragoussis J. Current and Future Methods for mRNA Analysis: A Drive Toward Single Molecule Sequencing. Methods Mol Biol. 2018;1783: 209–241. doi:10.1007/978-1-4939-7834-2_11

30. Griffith M, Walker JR, Spies NC, Ainscough BJ, Griffith OL. Informatics for RNA Sequencing: A Web Resource for Analysis on the Cloud. PLoS Comput Biol. 2015;11: e1004393. doi:10.1371/journal.pcbi.1004393

31. Shanker S, Paulson A, Edenberg HJ, Peak A, Perera A, Alekseyev YO, et al. Evaluation of commercially available RNA amplification kits for RNA sequencing using very low input amounts of total RNA. J Biomol Tech. 2015;26: 4–18. doi:10.7171/jbt.15-2601-001

32. Stegle O, Teichmann SA, Marioni JC. Computational and analytical challenges in single-cell transcriptomics. Nat Rev Genet. 2015;16: 133–145. doi:10.1038/nrg3833

33. Hwang B, Lee JH, Bang D. Single-cell RNA sequencing technologies and bioinformatics pipelines. Exp Mol Med. 2018;50: 96. doi:10.1038/s12276-018-0071-8

34. Li L, Clevers H. Coexistence of quiescent and active adult stem cells in mammals. Science. 2010;327: 542–545. doi:10.1126/science.1180794

35. Huang S. Non-genetic heterogeneity of cells in development: more than just noise. Development. 2009;136: 3853–3862. doi:10.1242/dev.035139

36. Shalek AK, Satija R, Shuga J, Trombetta JJ, Gennert D, Lu D, et al. Single-cell RNA-seq reveals dynamic paracrine control of cellular variation. Nature. 2014;510: 363–369. doi:10.1038/nature13437

37. Julius MH, Masuda T, Herzenberg LA. Pillars article: demonstration that antigen-binding cells are precursors of antibody-producing cells after purification with a fluorescence-activated cell sorter. Proc. Natl. Acad. Sci. 1972. 69: 1934-1938. J Immunol. 2014;193: 2048–2052. Available: https://www.ncbi.nlm.nih.gov/pubmed/25128550

38. Whitesides GM. The origins and the future of microfluidics. Nature. 2006;442: 368–373. doi:10.1038/nature05058

39. Lun ATL, McCarthy DJ, Marioni JC. A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor. F1000Research. 2016. p. 2122. doi:10.12688/f1000research.9501.2

40. Brennecke P, Anders S, Kim JK, Kołodziejczyk AA, Zhang X, Proserpio V, et al. Accounting for technical noise in single-cell RNA-seq experiments. Nat Methods. 2013;10: 1093–1095. doi:10.1038/nmeth.2645

41. Marinov GK, Williams BA, McCue K, Schroth GP, Gertz J, Myers RM, et al. From single-cell to cell-pool transcriptomes: stochasticity in gene expression and RNA splicing. Genome Res. 2014;24: 496–510. doi:10.1101/gr.161034.113

42. Ilicic T, Kim JK, Kolodziejczyk AA, Bagger FO, McCarthy DJ, Marioni JC, et al. Classification of low quality cells from single-cell RNA-seq data. Genome Biol. 2016;17: 29. doi:10.1186/s13059-016-0888-1

43. Haghverdi L, Buettner F, Theis FJ. Diffusion maps for high-dimensional single-cell analysis of differentiation data. Bioinformatics. 2015;31: 2989–2998. doi:10.1093/bioinformatics/btv325

44. Haghverdi L, Büttner M, Wolf FA, Buettner F, Theis FJ. Diffusion pseudotime robustly reconstructs branching cellular lineages. Nat Methods. 2016;412: 13.

45. Berggård T, Linse S, James P. Methods for the detection and analysis of protein–protein interactions. Proteomics. 2007;7: 2833–2842. doi:10.1002/pmic.200700131

46. Keskin O, Tuncbag N, Gursoy A. Predicting Protein-Protein Interactions from the Molecular to the Proteome Level. Chem Rev. 2016;116: 4884–4909. doi:10.1021/acs.chemrev.5b00683

47. Mercatelli D, Scalambra L, Triboli L, Ray F, Giorgi FM. Gene regulatory network inference resources: A practical overview. Biochim Biophys Acta Gene Regul Mech. 2020;1863: 194430. doi:10.1016/j.bbagrm.2019.194430

48. Kanehisa M. KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Research. 2000. pp. 27–30. doi:10.1093/nar/28.1.27

49. Slenter DN, Kutmon M, Hanspers K, Riutta A, Windsor J, Nunes N, et al. WikiPathways: a multifaceted pathway database bridging metabolomics to other omics research. Nucleic Acids Res. 2018;46: D661–D667. doi:10.1093/nar/gkx1064

50. Yu D, Kim M, Xiao G, Hwang TH. Review of biological network data and its applications. Genomics Inform. 2013;11: 200–210. doi:10.5808/GI.2013.11.4.200

51. Venkatesan K, Rual J-F, Vazquez A, Stelzl U, Lemmens I, Hirozane-Kishikawa T, et al. An empirical framework for binary interactome mapping. Nat Methods. 2009;6: 83–90. doi:10.1038/nmeth.1280

52. Murakami Y, Tripathi LP, Prathipati P, Mizuguchi K. Network analysis and in silico prediction of protein–protein interactions with applications in drug discovery. Curr Opin Struct Biol. 2017;44: 134–142. doi:10.1016/j.sbi.2017.02.005

53. Ruohonen K. Graph Theory. Tampereen teknillinen yliopisto. Originally titled Graafiteoria, lecture notes translated by Tamminen, J., Lee, K. C and Piché, R. 2013.

54. Wiwie C, Kuznetsova I, Mostafa A, Rauch A, Haakonsson A, Barrio-Hernandez I, et al. Time-Resolved Systems Medicine Reveals Viral Infection-Modulating Host Targets. Syst Med (New Rochelle). 2019;2: 1–9. doi:10.1089/sysm.2018.0013

55. Szklarczyk D, Franceschini A, Wyder S, Forslund K, Heller D, Huerta-Cepas J, et al. STRING v10: protein–protein interaction networks, integrated over the tree of life. Nucleic Acids Res. 2015;43: D447–D452. doi:10.1093/nar/gku1003

56. Schaefer MH, Serrano L, Andrade-Navarro MA. Correcting for the study bias associated with protein–protein interaction measurements reveals differences between protein degree distributions from different cancer types. Frontiers in Genetics. 2015. doi:10.3389/fgene.2015.00260

57. Jensen LJ, Bork P. Not Comparable, But Complementary. Science. 2008;322: 56–57. doi:10.1126/science.1164801

58. Björklund ÅK, Light S, Hedin L, Elofsson A. Quantitative assessment of the structural bias in protein--protein interaction assays. Proteomics. 2008;8: 4657–4667.

59. Kotlyar M, Pastrello C, Pivetta F, Lo Sardo A, Cumbaa C, Li H, et al. In silico prediction of physical protein interactions and characterization of interactome orphans. Nat Methods. 2015;12: 79–84. doi:10.1038/nmeth.3178

60. Rolland T, Taşan M, Charloteaux B, Pevzner SJ, Zhong Q, Sahni N, et al. A proteome-scale map of the human interactome network. Cell. 2014;159: 1212–1226. doi:10.1016/j.cell.2014.10.050

61. Oughtred R, Stark C, Breitkreutz B-J, Rust J, Boucher L, Chang C, et al. The BioGRID interaction database: 2019 update. Nucleic Acids Res. 2019;47: D529–D541. doi:10.1093/nar/gky1079

62 Delgado FM, Gómez-Vela F. Computational methods for Gene Regulatory Networks reconstruction and analysis: A review. Artificial Intelligence in Medicine. 2019. pp. 133–145. doi:10.1016/j.artmed.2018.10.006

63 Usadel B, Obayashi T, Mutwil M, Giorgi FM, Bassel GW, Tanimoto M, et al. Co-expression tools for plant biology: opportunities for hypothesis generation and caveats. Plant, Cell & Environment. 2009. pp. 1633–1651. doi:10.1111/j.1365-3040.2009.02040.x

64 Jia B, Xu S, Xiao G, Lamba V, Liang F. Learning gene regulatory networks from next generation sequencing data. Biometrics. 2017;73: 1221–1230. doi:10.1111/biom.12682

65 Giorgi FM, Del Fabbro C, Licausi F. Comparative study of RNA-seq- and microarray-derived coexpression networks in Arabidopsis thaliana. Bioinformatics. 2013;29: 717–724. doi:10.1093/bioinformatics/btt053

66 Aldrich J. Correlations Genuine and Spurious in Pearson and Yule. Statistical Science. 1995. pp. 364–376. doi:10.1214/ss/1177009870

67 Inukai S, Kock KH, Bulyk ML. Transcription factor–DNA binding: beyond binding site motifs. Current Opinion in Genetics & Development. 2017. pp. 110–119. doi:10.1016/j.gde.2017.02.007

68 Pavesi G. ChIP-Seq Data Analysis to Define Transcriptional Regulatory Networks. Advances in Biochemical Engineering/Biotechnology. 2016. pp. 1–14. doi:10.1007/10_2016_43

69 Lee TI, Young RA. Transcriptional Regulation and Its Misregulation in Disease. Cell. 2013. pp. 1237–1251. doi:10.1016/j.cell.2013.02.014

70. Goodfellow I, Bengio Y, Courville A. Introduction. Deep Learning. MIT Press; 2016. pp. 2–26. Available: https://www.deeplearningbook.org/contents/intro.html

71. Abiodun OI, Jantan A, Omolara AE, Dada KV, Mohamed NA, Arshad H. State-of-the-art in artificial neural network applications: A survey. Heliyon. 2018;4: e00938. doi:10.1016/j.heliyon.2018.e00938

72. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521: 436–444. doi:10.1038/nature14539

73. Nielsen MA. What this book is about. Neural networks and deep learning. Determination press San Francisco, CA, USA:; 2015. Available: http://neuralnetworksanddeeplearning.com/about.html

74. Sønderby SK, Sønderby CK, Nielsen H, Winther O. Convolutional LSTM Networks for Subcellular Localization of Proteins. Algorithms for Computational Biology. 2015. pp. 68–80. doi:10.1007/978-3-319-21233-3_6

75. Graves A. Supervised Sequence Labelling. In: Graves A, editor. Supervised Sequence Labelling with Recurrent Neural Networks. Berlin, Heidelberg: Springer Berlin Heidelberg; 2012. pp. 5–13. doi:10.1007/978-3-642-24797-2_2

76. Nielsen MA. Using neural nets to recognize handwritten digits. Neural networks and deep learning. Determination press San Francisco, CA, USA:; 2015. Available: http://neuralnetworksanddeeplearning.com/chap1.html

77. Goodfellow I, Bengio Y, Courville A. Deep Feedforward Networks. Deep learning. MIT press; 2016. pp. 164–223. Available: https://www.deeplearningbook.org/contents/mlp.html

78. Hinton GE, Osindero S, Teh Y-W. A fast learning algorithm for deep belief nets. Neural Comput. 2006;18: 1527–1554. doi:10.1162/neco.2006.18.7.1527

79. Yoshua Bengio YL. Scaling Learning Algorithms toward AI. Large Scale Kernel Machines. MIT Press; 2007. doi:10.7551/mitpress/7496.003.0016

80 LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, et al. Backpropagation Applied to Handwritten Zip Code Recognition. Neural Computation. 1989. pp. 541–551. doi:10.1162/neco.1989.1.4.541

81. Lecun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE. 1998;86: 2278–2324. doi:10.1109/5.726791

82. Hubel DH, Wiesel TN. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. The Journal of Physiology. 1962. pp. 106–154. doi:10.1113/jphysiol.1962.sp006837

83. Goodfellow I, Bengio Y, Courville A. Convolutional Networks. Deep Learning. MIT Press; 2016. pp. 326–366. Available: https://www.deeplearningbook.org/contents/convnets.html

84 Zhou, Zhou, Chellappa. Computation of optical flow using a neural network. IEEE International Conference on Neural Networks. 1988. doi:10.1109/icnn.1988.23914

85. Srivastava RK, Masci J, Kazerounian S, Gomez F, Schmidhuber J. Compete to Compute. In: Burges CJC, Bottou L, Welling M, Ghahramani Z, Weinberger KQ, editors. Advances in Neural Information Processing Systems 26. Curran Associates, Inc.; 2013. pp. 2310–2318.

86. Bahdanau D, Cho K, Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv [cs.CL]. 2014. Available: http://arxiv.org/abs/1409.0473

87. Schuster M, Paliwal KK. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing. 1997. pp. 2673–2681. doi:10.1109/78.650093

88. Hochreiter S, Bengio Y, Frasconi P, Schmidhuber J, Others. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. A field guide to dynamical recurrent neural networks. IEEE Press; 2001. Available: https://ml.jku.at/publications/older/ch7.pdf

89. Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. IEEE Trans Neural Netw. 1994;5: 157–166. doi:10.1109/72.279181

90. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9: 1735–1780. doi:10.1162/neco.1997.9.8.1735

91. Hastie T, Tibshirani R, Friedman J. Model Assessment and Selection. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition. Springer Science & Business Media; 2009. pp. 238–279.

92. Nielsen MA. Improving the way neural networks learn. Neural networks and deep learning. Determination press San Francisco, CA, USA:; 2015. Available: http://neuralnetworksanddeeplearning.com/chap3.html

93. Goodfellow I, Bengio Y, Courville A. Numerical Computation. Deep learning. MIT press; 2016. pp. 78–95. Available: https://www.deeplearningbook.org/contents/numerical.html

94. Goodfellow I, Bengio Y, Courville A. Practical Methodology. Deep learning. MIT press; 2016. pp. 416–437. Available: https://www.deeplearningbook.org/contents/guidelines.html

95. Werbos PJ. Backpropagation through time: what it does and how to do it. Proceedings of the IEEE. 1990. pp. 1550–1560. doi:10.1109/5.58337

96 Goodfellow I, Bengio Y, Courville A. Regularization for Deep Learning. Deep learning. MIT press; 2016. pp. 224–270. Available: https://www.deeplearningbook.org/contents/regularization.html

97 Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving neural networks by preventing co-adaptation of feature detectors. arXiv [cs.NE]. 2012. Available: http://arxiv.org/abs/1207.0580

98. Jain AK. Data clustering: 50 years beyond K-means. Pattern Recognition Letters. 2010. pp. 651–666. doi:10.1016/j.patrec.2009.09.011

99. Podani J. Hierarchical clustering. Introduction to the exploration of multivariate biological data. Backhuys Publishers; 2000. pp. 135–174. Available: http://ramet.elte.hu/~podani/subindex.html

100. Manning CD, Raghavan P, Schütze H. Hierarchical clustering. Introduction to Information Retrieval. USA: Cambridge University Press; 2008. pp. 377–401. Available: https://nlp.stanford.edu/IR-book/pdf/17hier.pdf

101. Podani J. Distance, similarity. Introduction to the exploration of multivariate biological data. Backhuys Publishers; 2000. pp. 55–110. Available: http://ramet.elte.hu/~podani/subindex.html

102. Podani J. Partitions. Introduction to the exploration of multivariate biological data. Backhuys Publishers; 2000. Pp 111–133 Available: http://ramet.elte.hu/~podani/subindex.html

103. Palit S, Heuser C, de Almeida GP, Theis FJ, Zielinski CE. Meeting the Challenges of High-Dimensional Single-Cell Data Analysis in Immunology. Front Immunol. 2019;10: 1515. doi:10.3389/fimmu.2019.01515

104. Aggarwal CC, Hinneburg A, Keim DA. On the Surprising Behavior of Distance Metrics in High Dimensional Space. Database Theory — ICDT 2001. 2001. pp. 420–434. doi:10.1007/3-540-44503-x_27

105. Dorrity MW, Saunders LM, Queitsch C, Fields S, Trapnell C. Dimensionality reduction by UMAP to visualize physical and genetic

interactions. Nat Commun. 2020;11: 1537. doi:10.1038/s41467-020-15351-4

106. Becht E, McInnes L, Healy J, Dutertre C-A, Kwok IWH, Ng LG, et al. Dimensionality reduction for visualizing single-cell data using UMAP. Nat Biotechnol. 2018. doi:10.1038/nbt.4314

107. McInnes L, Healy J, Melville J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv [stat.ML]. 2018. Available: http://arxiv.org/abs/1802.03426

108. Shlens J. A Tutorial on Principal Component Analysis. arXiv [cs.LG]. 2014. Available: http://arxiv.org/abs/1404.1100

109. How UMAP Works — umap 0.4 documentation. [cited 26 Apr 2020]. Available:https://umap-learn.readthedocs.io/en/latest/how_umap_works.html

110. Manifold - Encyclopedia of Mathematics. [cited 26 Apr 2020]. Available: https://www.encyclopediaofmath.org/index.php/Manifold

111. Topology, general - Encyclopedia of Mathematics. [cited 26 Apr 2020].Available:https://www.encyclopediaofmath.org/index.php/Topology,_general

2 Manuscript 1

Comparative single-cell trajectory network enrichment identifies mechanistic patterns in hematopoiesis and CD8 T-cell development

Comparative single-cell trajectory network enrichment identifies mechanistic patterns in hematopoiesis and CD8 T-cell development

Alexander G. B. Grønning[1], Mhaned Oubounyt[2], Kristiyan Kanev[3], Jesper Lund[4], Tim Kacprowski[5], Dietmar Zehn[3], Richard Röttger[1], Jan Baumbach[1,2]

[1] Department of Mathematics and Computer Science,University of Southern Denmark, 5000 Odense M, Denmark

[2] Chair of Experimental Bioinformatics, TUM School of Life Sciences Weihenstephan, Technical University of Munich, Freising, Germany

[3] Division of Animal Physiology and Immunology, TUM School of Life Sciences Weihenstephan, Technical University of Munich, 85354 Freising, Germany

[4] Department of Biostatistics and Epidemiology, University of Southern Denmark, 5000 Odense C, Denmark

[5] Research Group on Computational Systems Medicine, Chair of Experimental Bioinformatics, TUM School of Life Sciences, Technical University of Munich, Freising, Germany

Abstract

Single-cell transcriptomics (scRNA-seq) technologies allow for investigating cellular differentiation processes with unprecedented resolution. While powerful software packages for scRNA-seq raw data analysis exist, systems biology-based measurements and tools for trajectory analysis are rare and typically difficult to handle. This hampers biological exploration and prevents researchers from gaining deeper insights into the molecular control of developmental processes. We report Scellnetor, a network-constraint time-series clustering algorithm, which is fast enough to be integrated into an interactive webtool. It allows user-friendly extraction of temporal differential gene expression network patterns and kinetics. It supports comparing two differentiation courses or two developmental trajectories on a systems biology level. Using well characterized experimental model systems, we demonstrate the capacity of Scellnetor as a hypothesis generator to identify putative mechanisms driving hematopoiesis or mechanistically interpretable subnetworks driving dysfunctional CD8 T-cell development in chronic infections. Altogether, Scellnetor is the unique method to allow for single-cell trajectory network enrichment which effectively lifts scRNA-seq data analysis to a systems biology level. It is available as an interactive online tool at https://exbio.wzw.tum.de/scellnetor/.

Introduction

Single-cell RNA sequencing (scRNA-seq) allows researchers to perform cellular developmental studies with a hitherto unseen fine granularity. Single-cell transcriptomes have paved the way

for novel discoveries in various biomedical fields by improving the understanding of how transcriptional profiles relate to cell phenotypes. A range of algorithms have been invented for clustering of scRNA-seq data and for inferring differentiation trajectories (1–3). Clustering assumes that single-cells can be divided into distinct groups whereas trajectory inference aims to arrange cells such that continuous phenotypes can be traced on a low dimensional cell map (4). Important examples of the latter include diffusion maps (5, 6) and pseudotemporal ordering of single-cells (3, 7). Both algorithms seek to position single-cells such that their coordinates reflect their developmental statuses in relation to the other cells. Additionally, several software packages have been developed for the entire analysis pipeline, from pre-processing to clustering and identification of differentially expressed genes. Scanpy (8), Seurat (9) and SINCERA (10) are examples of such software packages. Though scRNA-seq data is still challenged by noise (11), combinations of different tools and algorithms have helped to unravel hidden inter-cellular mechanisms and shed light on unknown cellular paths of differentiation and disease progression (12–14).

Typical computational analyses of single-cell gene expression data involve a pre-processing step, where, e.g., cells with high levels of mitochondrial DNA and few expressed genes are removed (2, 15–18). This is followed by a clustering of single-cells' transcriptional profiles and/or inference of developmental trajectories (19, 20). Normally, the clusters or trajectory segments are validated and identified using expression of marker genes (11–14, 17, 21, 22). A way to examine development trajectories more mechanistically is by inferring gene regulatory networks from the scRNA-seq data in question (11, 17, 22–27). Despite being useful in specific scenarios, such (pseudo) gene regulatory networks are limited in their power to describe the interactome beyond transcription factors. The mechanistic patterns they describe do not grant a view on the full picture of the complexity of cellular developments. In bulk RNA-seq data analysis, network enrichment technology (e.g. KeyPathwayMiner (28), GiGa (29) or ActiveModules (30)) is typically applied to find such mechanistic patterns. However, these methods have not been developed to consider the noise of scRNA-seq data and to work with the computationally inferred pseudotime relationships of cells, making it difficult to get meaningful results when applying them to single-cell expression data.

Even though the standard approaches to scRNA-seq analysis outlined above can help to deduce new insights from scRNA-seq, they cannot (or any other tool) identify mechanistic patterns that explain cellular developmental programs over time at interactome-level. Moreover, no tools exist that can identify molecular subnetworks enriched with genes that are differently expressed in two distinct differentiation trajectories, thus unraveling how interacting genes might "push" cellular development in different directions. From a systems medicine point of view, as no tool for direct comparison between healthy differentiation trajectories and disease-associated development trajectories exists, it remains impossible to locate genes that in synergy, as a mechanism, are responsible for disease progression.

To fill this gap, we have invented Scellnetor: Single-cell Network Profiler for Extraction of Systems Biology Patterns from scRNA-seq Trajectories. The first (pseudo)temporal scRNA-seq network enrichment technique that can unravel subnetworks of genes crucial for explaining the progression of single cell development trajectories. The method allows users to compare single-cell trajectories selected on low-dimensional cell maps. The selected cell sets are pseudotime-

sorted and clustered using a hierarchical clustering algorithm that is constrained by the protein-protein interaction (PPI) network from BioGrid (31) (or a user-chosen network). Scellnetor identifies gene modules (connected subnetworks of genes) that are either differently or similarly expressed in the two selected sets of cells (Fig. 1). The tool is, therefore, able to extract mechanisms that are fundamental cellular driver programs for differentiating between distinct development courses. It is available as an intuitive and interactive online tool at https://exbio.wzw.tum.de/scellnetor.

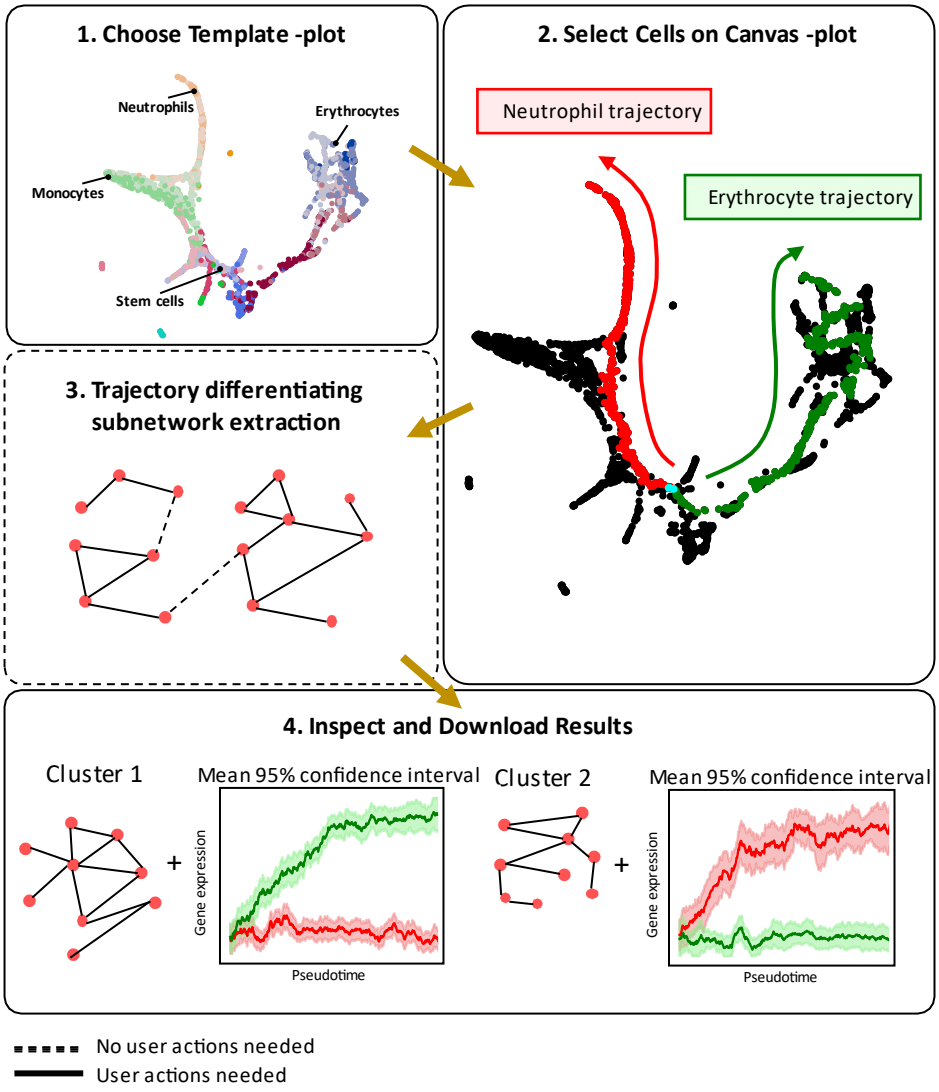


Figure 1. Workflow of Scellnetor. **1** Choosing desired Template-plot. **2** Selection of cells on Canvas-plot that represents differentiation trajectories. **3** Scellnetor performs a constrained hierarchical clustering based on expression data from the selected cells. **4** User can inspect and download the results from the Scellnetor clustering. Scellnetor outputs subnetworks of genes and mean gene expression with 95% confidence intervals.

Results

Differences between neutrophil and erythrocyte development trajectories

To validate the Scellnetor methodology, we used scRNA-seq data from (21). In their article, Paul *et al.* analyzed gene expression patterns of mouse hematopoietic cells while they differentiated to progenies from a pool of progenitor cells: common myeloid progenitors (CMPs), granulocyte-macrophage progenitors (GMPs), and megakaryocyte-erythrocyte progenitors (MEPs). Using an Expectation Maximization-based clustering approach, Paul *et al.* divided the single-cells into 19 different groups. Finally, they constructed a detailed map of the dynamic transcriptional states within the myeloid progenitor populations. Using the single-cells from the 19 clusters and their scRNA-seq gene expression data set (GSE72857) we constructed an ANNDATA object, created cell maps and computed pseudotime. Our ANNDATA object contained 2730 single-cells that expressed 3451 genes (see Method section for more info). Our pseudotime calculation was based on progenitor clusters defined by Paul *et al.* (2016) (cluster 7-11 in Fig. S1a) and coordinates from a cell map based on principles of force-directed graph drawing (32), which showed clear branching of the differentiated cells. Also, the plot showed clear co-localization of Paul *et al.* clusters that were highly similar. The "start cell" for the pseudotime computation (7) was the cell closest to the average position of all relevant progenitor clusters (cluster 7-10 in Fig. S1a, Fig. S1b). The cluster 11 was omitted, as it was dislocated from the remainder of the cells on the plot (Fig. S1a). The connected area where Paul *et al.* clusters 7-10 are co-localized will be referred to as "stem cell area" (Fig. S1a, Fig. S1b).

We ran our Scellnetor algorithms to extract comparative systems biology profiles between 1.) the trajectory from stem cells towards differentiated neutrophils versus 2.) the trajectory from stem cells towards differentiated erythrocytes (Fig. 2a). The drawn paths go through several of the Paul *et al.* (2016) defined clusters. Outside the stem cell area, the neutrophil path goes through clusters 15-17, where 16-17 are the neutrophil clusters. The erythrocyte path goes through cells in the stem cell area and the clusters 1-6, which all are erythrocyte clusters (Fig. 2a, Fig. S1a). We identified seven clusters with a minimum size of five genes using pseudotime as the sorting metric. The distance metric was Euclidean, the linkage type was complete and the size of the moving average was 20. Using published data, we extracted single-cells from differentiation trajectories and clustered them in a comparative analysis.

Pseudo-timelines and validation

In Figure 2, we show three of the clusters. Figure 2b, d, and f show plots of the mean and 95% confidence intervals of the smoothed average expression in the three clusters (see supplementary method section). Below the plots are tables with GO-terms linked to each cluster (see next section). The red lines on the plots indicate the expression values of the clustered genes when following the path from stem cells to neutrophils while the green lines correspond to expression on the path from stem cell to erythrocytes. One can see how the modules' genes' expression differs over pseudotime between the neutrophil and erythrocyte trajectories. Using Wilcoxon signed-rank test, we show that the average expression over time of the clustered genes from the

two distinct differentiation paths are statistically significantly different (q-value $2.52 \cdot 10^{-59}$ for all clusters).

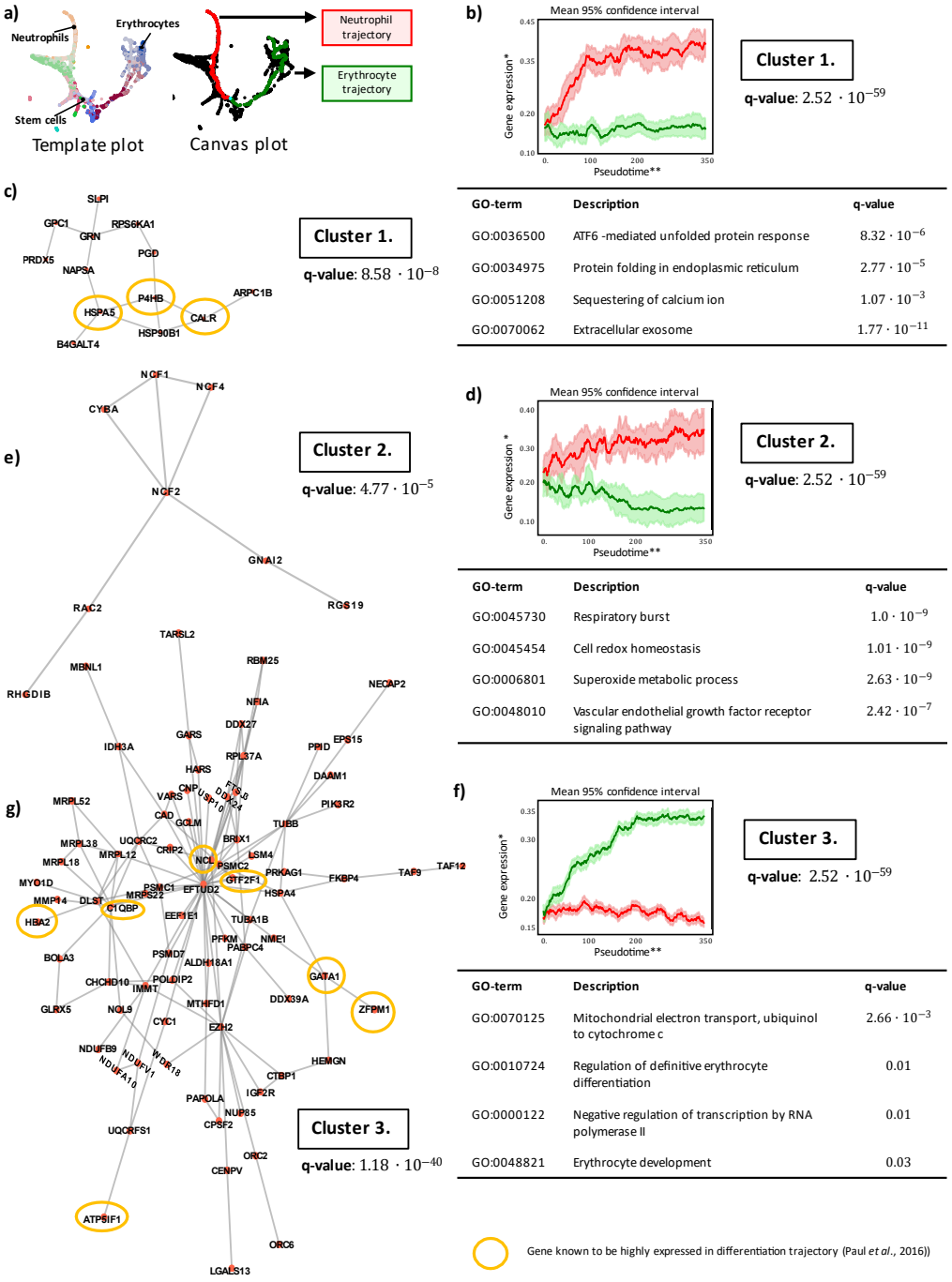


Figure 2. Comparison of neutrophil and erythrocyte differentiation trajectories. **a** Chosen Template-plot (left) and selected cells on Canvas-plot (right). The selected cells form paths representing trajectories from stem cells towards differentiated neutrophils and from stem cells towards differentiated erythrocytes, respectively. **b, d, f** Plots of mean pseudo-timelines and 95% confidence interval of cluster 1, 2 and 3. The mean pseudo-timelines from both trajectories of each cluster were compared using Wilcoxon signed-rank tests. Resulting q-values can be seen below every plot (p-values are adjusted using Benjamini-Hochberg Procedure). Below every plot are tables with four statistically significant GO-terms linked to the clusters. **c, e, g** Gene modules corresponding to cluster 1, 2 and 3. The encircled genes are known to be highly expressed in the differentiation trajectory that on average has the highest expression in **b, d, f**(21). The associated q-values are based on Mann-Whitney *U* tests and indicate that these clusters are statistically significantly different from randomly generated clusters (p-values are adjusted using Benjamini-Hochberg Procedure). *Gene expression = Normalized moving-average-modified gene expression, **Pseudotime = Moving-average-modified number of single-cells.

Gene expression of cells in Scellnetor paths is consistent with gene expression in pre-defined clusters

In Figure 2a, the neutrophil path passes through stem cells and cells from Paul *et al.* clusters 15-17, whereas the erythrocyte path goes through stem cells and cells from Paul *et al.* clusters 1-6 . The paths contain subsets of these clusters, so it was of interest to check if the differences in expression between the paths were consistent with the differences in expression between the corresponding clusters (e.g. clusters 15-17 vs. clusters 1-6 according to (21)). We extracted all genes from the Scellnetor clusters (Fig. 2, Fig. S2) and created two cell sets; one containing all cells from the Paul *et al.* clusters 15-17 and one containing all cells from the Paul *et al.* clusters 1-6. Based on the two sets, we used our initial count matrix (see method section) to compute the average expression of the genes in the Scellnetor clusters. We plotted the averaged gene expression values and compared the resulting distributions using Wilcoxon signed-rank test (Fig. S3a-g). The same was done for the subsets of the above Paul *et al.* clusters that were included in the paths shown in Figure 2a (Fig. S3h-n). For these calculations, we used the pre-processed expression matrix from our ANNDATA object (see method section). We found that the cells in the Scellnetor paths (Fig. 2a), expressed genes in a manner that was consistent with the gene expression of the cells in the relevant Paul *et al.* clusters.

Scellnetor clusters point towards erythropoiesis

The clustered genes as connected subnetworks can be seen on Figure 2c, e, g. The q-values next to every subgraph are based on Mann-Whitney *U* tests and indicate that these clusters are statistically significantly different from randomly generated clusters (q-values for cluster 1, 2, 3 are $8.58\cdot 10^{-8}$, $4.77\cdot 10^{-5}$, $1.18\cdot 10^{-40}$, respectively). The genes *P4HB*, *CALR* and *HSPA5* in cluster 1 (Fig. 2c) produce surface markers that are expressed at high levels in neutrophils. The genes *GATA1*, *ZFPM1* and *GTF2F1* in cluster 3 (Fig. 2g) code for transcription factors that are upregulated in erythrocyte differentiating cells. The genes *NCL*, *HBA2*, *C1QBP* and *ATP5IF1* are all known marker genes associated with the erythrocyte lineage (Fig. 2g). Additional transcription factors upregulated in erythropoiesis are *GFI1B, LMO2* and *CBFA2T3*, which were found in Scellentor cluster 7 (Figure S2h) (21). The genes in the Scellentor cluster 3 (Fig. 2g) are more highly expressed on average in the cells located in the erythrocyte trajectory. This is corroborated by higher expression of *HBA2*, which codes for a subunit of hemoglobin (33) and *ATP5IF1*, which codes for a mitochondrial ATPase inhibitor that is involved in the in the synthesis of hemoglobin (34).

Gene Set Enrichment Analysis Results

Scellnetor can automatically conduct gene set enrichment analysis when gene modules have been identified. To demonstrate this functionality, we display four biologically meaningful statistically significant biological process GO-terms associated with each cluster in the tables in Figure 2b, d, f. To see all GO-terms associated with the clusters, see supplementary file 1. Genes involved in “ATF6-mediated unfolded protein response”, “protein folding in endoplasmic reticulum” and “sequestering of calcium ion” have been found highly expressed in cluster 1 (Fig. 2b). The first two GO-terms describe events that have to do with protein folding and

endoplasmic reticulum (ER) stress. It has been shown that ER stress is decreased during neutrophil differentiation (35). Scellnetor uncovered that *CALR* is highly expressed in the neutrophil trajectory compared to the erythrocyte trajectory. *CALR* code for a multi-functional protein called calreticulin, which counteracts ER stress and is involved in correct maintenance of calcium ions in the ER (36–40). The gene *P4HB* (Fig. 2e) have also been found to downregulate ER stress (41). Eleven out of 13 genes in cluster 1 (Fig. 2b) (except for *B4GALT4* and *RPS6KA1*) are associated with the GO-term "extracellular exosome". Releasing exosomes is an important part of neutrophil signaling (42). The GO-terms "respiratory burst", "cell redox homeostasis" and "superoxide metabolic process" from cluster 2 (Fig. 2d) all relate to well-known neutrophilic cellular processes. A distinctive feature of the inflammatory actions of neutrophils is the respiratory or oxidative burst, where large amounts of oxygen are consumed to produce superoxide (43). Recent studies revealed that neutrophils might play a key role in angiogenesis, as they can store and synthesize molecules with known angiogenic activity (fourth GO-term, Fig. 2d) (44–47). Cluster 2 does not contain any genes that were highlighted by (21) as noteworthy for the neutrophilic differentiation course. Still, it has been described that the neutrophil cytosolic factor 1, 2, 4 (*NCF1, NCF2, NCF4*) interact with *CYBA* and *RAC2* at the membrane as subunits of the NOX2 complex. Interestingly, the inclusion of *RAC2* in this molecular assembly is specific for neutrophils (48–50).

The GO-terms "regulation of definitive erythrocyte differentiation" and "erythrocyte development" from cluster 3 (Fig. 2f) indicate that the cells from the erythrocyte trajectory in fact do express genes associated with erythropoiesis. The GO-term "mitochondrial electron transport, ubiquinol to cytochrome c" (Fig. 2f) is interesting, because the proteins resulting from the genes linked to this term (*UQCRC2, UQCRFS1, CYC1*, Fig. 2g) have been found in high abundance in erythroid progenitor cells compared to hematopoietic stem cells (51). Apparently, these genes are higher expressed in erythrocyte differentiating cells than in neutrophils differentiating cells. The GO-term "negative regulation of RNA polymerase II" fits well with genes involved in erythropoiesis, since mature erythrocytes lose their nuclei, which is where RNA polymerase II catalyze transcription of genes to pre-mRNA (52).

Scellnetor identifies trajectory-explaining gene modules for functional and dysfunctional exhausted CD8 T cells in chronic infection

To demonstrate that Scellnetor can be used for identifying mechanisms underlying disease progression, we reanalysed a dataset of pathways of CD8 T cell differentiation in a well-defined standard model of chronic infections (12). In the original work, the authors investigated proliferation and differentiation of CD8 T-cell progenitors to effector cells in populations of dysfunctional and normal CD8 T-cells with and without CD4 T-cell help. They revealed that absence of CD4 T-cell reduces the number of terminally differentiated CD8 T-cells, but not the number of CD8 T-cell progenitors in exhausted T-cell populations. Thus, the progenitor cells can maintain their population size without CD4 T-cell help. They identified 5 clusters and labeled them based on previously established signature genes; cluster 1 represented the critical stem-like progenitors, cluster 2 the functional and clusters 3, 4 and 5 the dysfunctional effector cells. In our study, we used the cells from these 5 clusters (GSE137007) to generate an

ANNDATA object, calculate pseudotime and compute a diffusion map (see method section). We used their cluster annotations to color code the cells (Fig. 3a).

We defined two trajectories of differentiation starting from the progenitor subpopulation (cluster 1), one towards the functional effector cells (cluster 2) and one towards the dysfunctional effector cells (cluster 3 and 4). We excluded cluster 5 (Figure 3a, cells in black) from the analysis as it is condition-specific and it was only observed in the absence of CD4 help. Using Scellnetor, we unravelled a gene module with increased expression in cells progressing in the dysfunctional trajectory and decreased expression in those progressing in the functional trajectory (Fig. 3b, c). The complexity of the gene module is highlighted by the diversity of genes and their functions, including processes like regulation of gene expression (epigenetic, transcriptional and translational), signalling, immune defence, cell migration, cytoskeletal reorganization and genes involved in the T cell receptor signalling pathway (*CD3G*, *FYN*, *ZAP70*, *LAT*, *ITK*, *PTPN22*, *LCP2*, *SLA2*, *NEDD4*, *NEDD9*, *LRRK1*, *SH2D2A* and *SH2D3C*), whose excessive stimulation is known to be one of the main drivers of T cell exhaustion (53). The two most statistically significant GO-terms associated with this cluster are "T-cell receptor signaling pathway" and "T-cell activation" (see supplementary file 2). TCR signalling is connected to two therapeutically interesting receptors involved in regulation of the T cell response – ILRD1 and TNFRSF9, which could potentially be used for its modulation. ILDR1 functions as regulator of T cells response in chronic infection and cancer is intriguing, especially taken into account that ILRD2 was recently described as a negative regulator of the T cells (54). However, up to now it has never been linked to controlling CD8 T cell function neither in chronic infection nor in cancer. TNFRSF9 (coding 4-1BB) is a positive regulator of T cell effector function and survival, and its stimulation has already been reported to ameliorate T cell exhaustion (55). The mechanistic gene module also includes diverse regulators of NF-κB signaling (PPP6R3, TERF2IP and EFHD2), which is critical for T cell survival and cytokine production (56). The negative regulator EFHD2 is of particular interest, as it is necessary for PD-1 mediated inhibition of proliferation and cytokine secretion in dysfunctional CD8 T cells (57).

As previously reported, the transition from functional to dysfunctional CD8 T cells is associated with metabolic reprogramming marked by a switch from glycolysis to oxidative phosphorylation (OXPHOS) as a main pathway for generation of adenosine triphosphate (ATP). In line with this, our dysfunctional gene module includes multiple genes involved in fatty-acid beta oxidation and OXPHOS (MDH1, KIAA1191, IDH3A, ECH1, OXCT1 and PRDX5) and putative regulators of this metabolic adaptation (HIF1AN, ARID5B and MTRF1L). Interestingly, HIF1AN is an inhibitor of HIF1α, which is known to trigger the expression of genes promoting the use of glycolysis over mitochondrial oxidative phosphorylation as main energy generating pathway (58–60). Moreover, HIFs activity has been shown to enhance the effector CD8 T cell response and influence the expression of pivotal transcription, effector and co-stimulatory molecules in chronic infection (61). Altogether, this demonstrates that the gene modules extracted by Scellnetor are functionally related mechanisms and precisely reflect the opposing nature of progenitor differentiation towards functional or dysfunctional CD8 T cells. Thus, Scellnetor is a promising systems medicine hypothesis generator identifying molecular sub-networks mechanistically driving dynamic differentiation of complex cell populations.

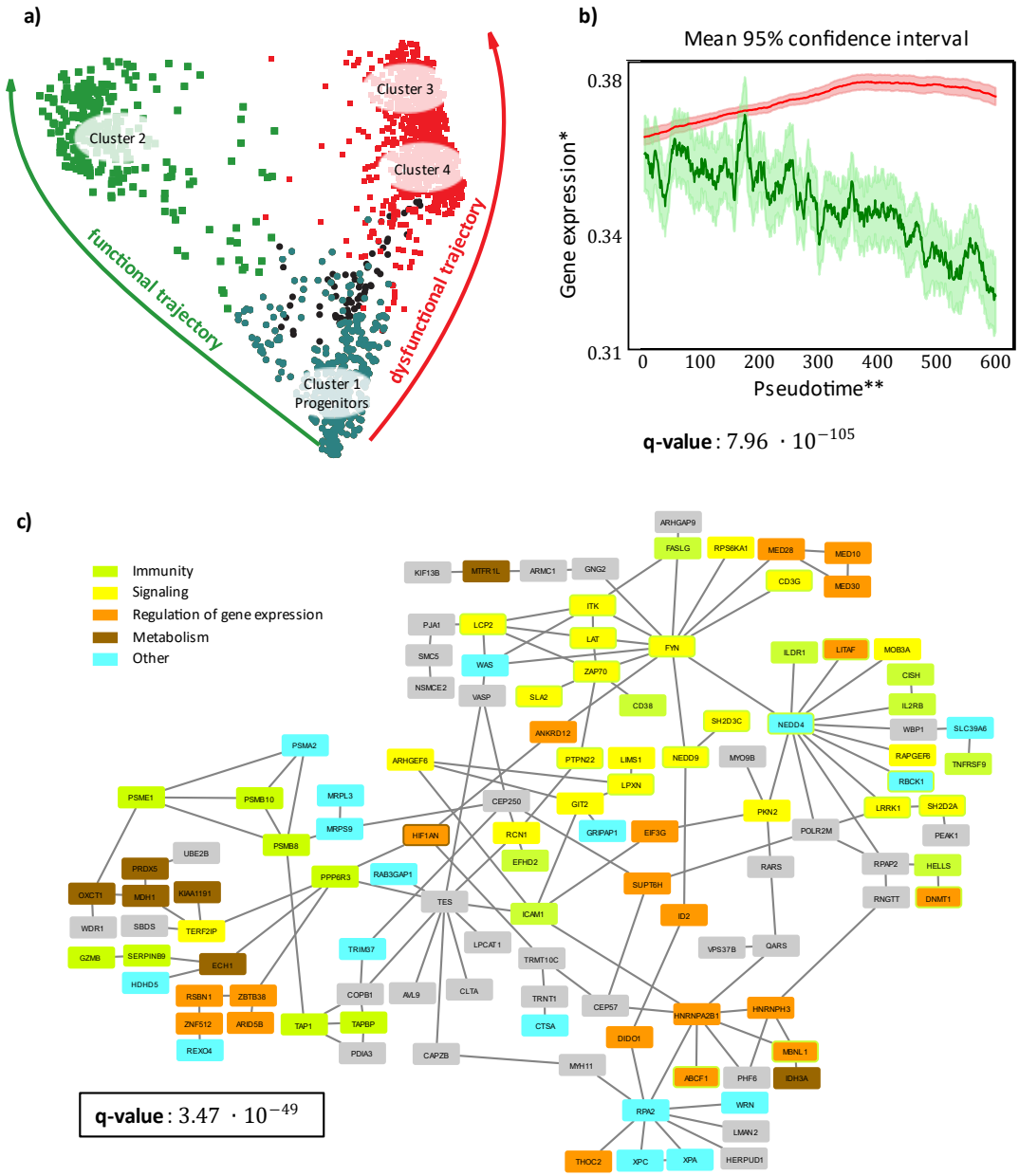


Figure 3. Differentiation of progenitor cells to dysfunctional CD8 T-cells. a Diffusion map of the single-cells as they differentiate to functional and dysfunctional CD8 T-cells. On the left, in green, is the development trajectory of cells from progenitor cells (cluster 1 in (12)) to functional CD8 T-cell (cluster 2 in (12)). On the right, in red, is the development trajectory of cells from progenitor cells to dysfunctional CD8 T-cell (cluster 3, cluster 4 in (12)). **b** Mean gene expression and 95% confidence intervals of the selected cluster. The genes in the cluster shown in **c** are, on average, expressed at a higher level in the cells developing via the functional trajectory. The q-value below the plot is based on a Wilcoxon signed-rank test comparing the average gene expressions. **c** The module induced by the cluster of genes. The genes are color coded; green genes are involved in immune responses, yellow genes are involved in signaling, orange genes are involved in regulation of gene expression, brown genes are involved in metabolism, blue genes are involved in other processes and grey genes do not belong to any of the mentioned groups. The color coding and, thereby, grouping of genes is based on our subjective assessments of their general functions. The associated q-value is based on Mann-Whitney *U* test. It indicates that the cluster is statistically significantly different from randomly generated clusters. All p-values in this figure are adjusted using Benjamini-Hochberg Procedure. *Gene expression = Normalized moving-average-modified gene expression, **Pseudotime = Moving-average-modified number of single-cells.

Discussion

Despite the wealth of different tools designated for the analysis of scRNA-seq raw data, *de novo* identification of interacting subnetworks expressed in a similar fashion across pseudotime has been neglected so far. With Scellnetor, we present the first algorithm (implemented as intuitive webtool and stand-alone software) for directly identifying modules of connected genes with similar expression patterns along pseudotime. It allows for the comparison of two user-selected trajectories and finding gene modules that are either similarly of differently expressed through pseudotime. It uncovers interacting genes that are involved in the cellular differentiation.

To showcase this, we used our novel tool to cluster scRNA-seq data from a hematopoiesis study by (21). We compared a developmental trajectory from stem cells towards differentiated neutrophils with a trajectory from stem cells towards differentiated erythrocytes. We uncovered statistically significant gene modules that are involved in differentiation of neutrophils and erythrocytes, respectively. The identified modules contained genes that previously have been associated with the two distinct differentiation courses and genes with GO-terms that were closely connected to the analyzed cellular developments (Fig. 2), demonstrating the biological relevance of the discovered modules.

We also applied Scellnetor to examine diseases progression by analyzing scRNA-seq data from (12). We compared two trajectories starting from the progenitor subpopulation (cluster 1 in (12)); one towards the functional effector cells (cluster 2 in (12)) and one towards the dysfunctional effector cells (cluster 3 and 4 in (12)) (Fig. 3). We found several statistically significant clusters, of which we highlight one module consisting of genes that are, on average, significantly higher expressed in cells developing in the dysfunctional trajectory than in cells developing along the path towards “healthy” effector cells (Fig. 3). This module contains many genes that have already been associated with dysfunctional CD8 T-cell development but also new genes that offer new insights into the mechanistic foundation of CD8 T-cell development.

To sum up, Scellnetor employs a novel network-constrained hierarchical agglomerative clustering algorithm, which enables the comparison of two sets of single-cells taking scRNA-seq data and a molecular interaction network as input. It is the first single-cell network enrichment method allowing the extraction of systems biology patterns from scRNA-seq trajectories. In two real-world data sets we demonstrated its potential to extract disease progression mechanisms and cellular programs driving cell differentiation.

Methods

Scellnetor is a novel algorithmic approach implemented into a responsive webtool that clusters scRNA-seq data (Fig. 4). It starts with a molecular interaction (usually: PPI) network, as well as a user-generated cell map from user-given scRNA-seq data. The coordinates of the cells are extracted and plotted on a canvas, where users may interactively select two trajectories or two sets of pre-defined clusters. Trajectories are selected by drawing paths on the canvas-plot, which are then connected by a greedy path-finding and smoothing algorithm. Clusters can be compared using subnetwork enrichment for differentiating expression module extraction. In contrast,

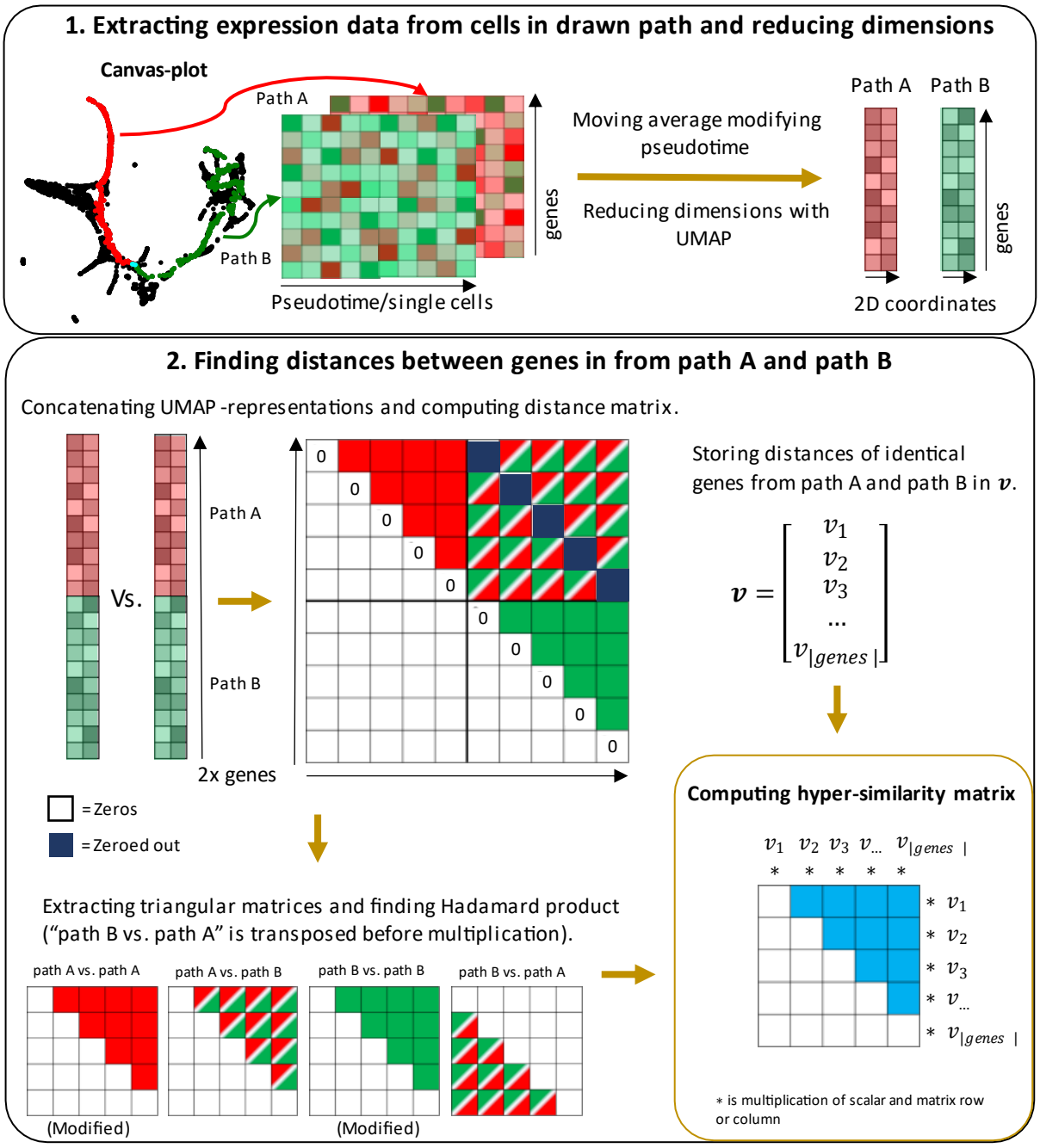


Figure 4. Computation of hyper-similarity matrix for gene module discovery. **1** Two paths are drawn on the canvas-plot, path A and path B. Expression matrices (one for each path) are created based on data from selected cells. All data that are derived from path A or path B will be labeled as path A or path B in the figure. In the matrices, rows are genes and columns are cells (pseudo-timepoints). Rows of the matrices are smoothed using a moving average function. The smoothed expression matrices are dimensionality reduced with UMAP. **2** The resulting 2D coordinate sets from path A and path B, respectively, are concatenated along the x-axes. The concatenated coordinate sets are used to produce a distance matrix, which is normalized by division of its max value. The diagonal, the lower triangular matrix and the diagonal of the upper right quadrant of the distance matrix are zeroed out. The distances between genes with identical IDs, but from different sets, are computed and stored in the vector, v (top, right). Triangular matrices of the distance matrix' quadrant I, II and IV (as in the cartesian coordinate system) are extracted. Values of the matrices "path A vs. path A" and "path B vs. path B" are converted into similarities. The matrix "path B vs. path A set" is transposed and the Hadamard product of all triangular matrices is computed (bottom, left). This produces the hyper-similarity matrix. The rows and columns of the hyper-similarity matrix are weighted by the values of v (bottom, right).

trajectory cells are sorted along pseudotime (7) to extract differentially expressed temporal subnetwork modules. Note that the webtool also allows identifying similarly (not differentially) expressed subnetwork modules, both for clusters and trajectories. Essentially, our network enrichment algorithm relies on computing a hyper-similarity matrix (genes vs. genes) that reflects the distance of pairs of genes in the network and their (dis)similarity of expression between the two selected clusters or over the two selected trajectories. This matrix is then clustered hierarchically and the induced subnetwork of emerging gene clusters are reported as candidate subnetwork modules. Scellnetor's output are 1) the corresponding subnetwork modules and the corresponding gene sets together with 2) plots of the mean expression development (over pseudotime) of the genes in the modules including the 95% confidence intervals, and 3) statistically significantly module-associated GO-terms. For a more detailed description of the Scellnetor methodology see supplementary method section and Fig. S4.

Author contribution statements

AG developed and implemented the clustering algorithm of the Scellnetor tool. AG developed and implemented all basic backend functionalities of the webtool. AG, JL and MO further developed the webtool and significantly contributed to debugging. KK, TK, and DZ tested the webtool, provided critical feedback, and together with AG used Scellnetor to generate the biomedical result presented in the manuscript. All authors equally contributed to writing and improving the paper. AG, JB and RR conceived the idea of the Scellnetor pipeline.

Software availability

Scellnetor is freely available as an online tool at https://exbio.wzw.tum.de/scellnetor/ and can be downloaded as standalone program from GitLab (https://gitlab.com/AlexTheKing/scellnetor_docker_public).

Data availability

The scRNA-seq data used for the Scellentor hematopoiesis clustering is from GEO (GSE72857). The scRNA-seq used for the clustering of exhausted CD8 T-cells in chronic infections is also from GEO (GSE137007). All Scellnetor results can be downloaded from GitLab (https://gitlab.com/AlexTheKing/scellnetor_docker_public).

Code availability

On GitLab (https://gitlab.com/AlexTheKing/scellnetor_docker_public) one may find the analysis scripts and data representations used for the preparation of the manuscript.

Acknowledgements

JB and AG received funding from JB's VILLUM Young Investigator Grant nr. 13154. The work of JB and TK was further funded by H2020 project RepoTrial (nr. 777111). The work of RR and JB has partially been funded by H2020 project FeatureCloud (nr. 826078). JB and TK grateful for financial support from BMBF project Sys_Care. MO is grateful for financial support of the Collaborative Research Center SFB924.

Conflict of interests

The authors declare not to have any conflict of interest.

References

1. V. A. Traag, L. Waltman, N. J. van Eck, From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.* **9**, 5233 (2019).

2. F. A. Wolf, *et al.*, PAGA: graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. *Genome Biology* **20** (2019).

3. L. Haghverdi, M. Büttner, F. A. Wolf, F. Buettner, F. J. Theis, Diffusion pseudotime robustly reconstructs branching cellular lineages. *Nat. Methods* **412**, 13 (2016).

4. V. Y. Kiselev, T. S. Andrews, M. Hemberg, Challenges in unsupervised clustering of single-cell RNA-seq data. *Nat. Rev. Genet.* **20**, 273–282 (2019).

5. R. R. Coifman, *et al.*, Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the National Academy of Sciences* **102**, 7426–7431 (2005).

6. L. Haghverdi, F. Buettner, F. J. Theis, Diffusion maps for high-dimensional single-cell analysis of differentiation data. *Bioinformatics* **31**, 2989–2998 (2015).

7. S. Tritschler, *et al.*, Concepts and limitations for learning developmental trajectories from single cell genomics. *Development* **146** (2019).

8. F. A. Wolf, P. Angerer, F. J. Theis, SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol.* **19**, 15 (2018).

9. A. Butler, P. Hoffman, P. Smibert, E. Papalexi, R. Satija, Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat. Biotechnol.* **36**, 411–420 (2018).

10. M. Guo, H. Wang, S. S. Potter, J. A. Whitsett, Y. Xu, SINCERA: A Pipeline for Single-Cell RNA-Seq Profiling Analysis. *PLoS Comput. Biol.* **11**, e1004575 (2015).

11. G. Chen, B. Ning, T. Shi, Single-Cell RNA-Seq Technologies and Related Computational Data Analysis. *Front. Genet.* **10**, 317 (2019).

12. K. Kanev, *et al.*, Proliferation-competent Tcf1+ CD8 T cells in dysfunctional populations are CD4 T cell help independent. *Proc. Natl. Acad. Sci. U. S. A.* **116**, 20070–20076 (2019).

13. X. Guo, *et al.*, Publisher Correction: Global characterization of T cells in non-small-cell lung cancer by single-cell sequencing. *Nat. Med.* **24**, 1628 (2018).

14. A. Sierksma, *et al.*, Novel Alzheimer risk genes determine the microglia response to amyloid-β but not to TAU pathology. *EMBO Mol. Med.* (2019).

15. C. Weinreb, S. Wolock, A. M. Klein, SPRING: a kinetic interface for visualizing high dimensional single-cell expression data. *Bioinformatics* **34**, 1246–1248 (2018).

16. R. Satija, J. A. Farrell, D. Gennert, A. F. Schier, A. Regev, Spatial reconstruction of single-cell gene expression data. *Nat. Biotechnol.* **33**, 495–502 (2015).

17. M. D. Luecken, F. J. Theis, Current best practices in single-cell RNA-seq analysis: a tutorial. *Mol. Syst. Biol.* **15** (2019).

18. B. Hwang, J. H. Lee, D. Bang, Single-cell RNA sequencing technologies and bioinformatics pipelines. *Exp. Mol. Med.* **50**, 96 (2018).

19. O. Stegle, S. A. Teichmann, J. C. Marioni, Computational and analytical challenges in single-cell transcriptomics. *Nat. Rev. Genet.* **16**, 133–145 (2015).

20. A. T. L. Lun, D. J. McCarthy, J. C. Marioni, A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor. *F1000Research* **5**, 2122 (2016).

21. F. Paul, *et al.*, Transcriptional Heterogeneity and Lineage Commitment in Myeloid Progenitors. *Cell* **164**, 325 (2016).

22. V. Moignard, *et al.*, Decoding the regulatory network of early blood development from single-cell gene expression measurements. *Nat. Biotechnol.* **33**, 269–276 (2015).

23. X. Fan, *et al.*, Spatial transcriptomic survey of human embryonic cerebral cortex by single-cell RNA-seq analysis. *Cell Res.* **28**, 730–745 (2018).

24. A. Wuidart, *et al.*, Early lineage segregation of multipotent embryonic mammary gland progenitors. *Nat. Cell Biol.* **20**, 666–676 (2018).

25. H. Matsumoto, *et al.*, SCODE: an efficient regulatory network inference algorithm from single-cell RNA-Seq during differentiation. *Bioinformatics* **33**, 2314–2321 (2017).

26. T. E. Chan, M. P. H. Stumpf, A. C. Babtie, Gene Regulatory Network Inference from Single-Cell Data Using Multivariate Information Measures. *Cell Syst* **5**, 251–267.e3 (2017).

27. S. Aibar, *et al.*, SCENIC: single-cell regulatory network inference and clustering. *Nat. Methods* **14**, 1083–1086 (2017).

28. N. Alcaraz, *et al.*, De novo pathway-based biomarker identification. *Nucleic Acids Res.* **45**, e151 (2017).

29. R. Breitling, A. Amtmann, P. Herzyk, Graph-based iterative Group Analysis enhances microarray interpretation. *BMC Bioinformatics* **5**, 100 (2004).

30. T. Ideker, O. Ozier, B. Schwikowski, A. F. Siegel, Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics* **18 Suppl 1**, S233–40 (2002).

31. R. Oughtred, *et al.*, The BioGRID interaction database: 2019 update. *Nucleic Acids Res.* **47**, D529–D541 (2019).

32. M. Jacomy, T. Venturini, S. Heymann, M. Bastian, ForceAtlas2, a continuous graph layout algorithm for handy network visualization designed for the Gephi software. *PLoS One* **9**, e98679 (2014).

33. D. M. Ribeiro, M. F. Sonati, Regulation of human alpha-globin gene expression and alpha-thalassemia. *Genet. Mol. Res.* **7**, 1045–1053 (2008).

34. D. I. Shah, *et al.*, Mitochondrial Atpif1 Regulates Heme Synthesis in Developing Erythroblasts. *Blood* **118**, 343–343 (2011).

35. A. Tanimura, *et al.*, Mitochondrial Activity and Unfolded Protein Response are Required for Neutrophil Differentiation. *Cell. Physiol. Biochem.* **47**, 1936–1950 (2018).

36. Y. Lim, *et al.*, Sumoylation regulates ER stress response by modulating calreticulin gene expression in XBP-1-dependent mode in Caenorhabditis elegans. *Int. J. Biochem. Cell Biol.* **53**, 399–408 (2014).

37. L. I. Gold, *et al.*, Calreticulin: non-endoplasmic reticulum functions in physiology and disease. *FASEB J.* **24**, 665–683 (2010).

38. T. Klampfl, *et al.*, Somatic mutations of calreticulin in myeloproliferative neoplasms. *N. Engl. J. Med.* **369**, 2379–2390 (2013).

39. M. Michalak, J. Groenendyk, E. Szabo, L. I. Gold, M. Opas, Calreticulin, a multi-process calcium-buffering chaperone of the endoplasmic reticulum. *Biochem. J* **417**, 651–666 (2009).

40. J. Nangalia, *et al.*, Somatic CALR mutations in myeloproliferative neoplasms with nonmutated JAK2. *N. Engl. J. Med.* **369**, 2391–2405 (2013).

41. S. Sun, *et al.*, Inhibition of prolyl 4-hydroxylase, beta polypeptide (P4HB) attenuates temozolomide resistance in malignant glioma via the endoplasmic reticulum stress response (ERSR) pathways. *Neuro-Oncology* **15**, 562–577 (2013).

42. A. Vargas, F. Roux-Dalvai, A. Droit, J.-P. Lavoie, Neutrophil-Derived Exosomes: A New Mechanism Contributing to Airway Smooth Muscle Remodeling. *American Journal of Respiratory Cell and Molecular Biology* **55**, 450–461 (2016).

43. C. C. Winterbourn, A. J. Kettle, M. B. Hampton, Reactive Oxygen Species and Neutrophil Function. *Annu. Rev. Biochem.* **85**, 765–792 (2016).

44. P. Scapini, *et al.*, CXCL1/macrophage inflammatory protein-2-induced angiogenesis in vivo is mediated by neutrophil-derived vascular endothelial growth factor-A. *J. Immunol.* **172**, 5034–5040 (2004).

45. M. A. Cassatella, Neutrophil-derived proteins: selling cytokines by the pound. *Adv. Immunol.* **73**, 369–509 (1999).

46. M. Gaudry, *et al.*, Intracellular pool of vascular endothelial growth factor in human neutrophils. *Blood* **90**, 4153–4161 (1997).

47. P. Scapini, F. Calzetti, M. A. Cassatella, On the detection of neutrophil-derived vascular endothelial growth factor (VEGF). *J. Immunol. Methods* **232**, 121–129 (1999).

48. C. O. Jacob, *et al.*, Lupus-associated causal mutation in neutrophil cytosolic factor 2 (NCF2) brings unique insights to the structure and function of NADPH oxidase. *Proc. Natl. Acad. Sci. U. S. A.* **109**, E59–67 (2012).

49. W. M. Nauseef, Assembly of the phagocyte NADPH oxidase. *Histochem. Cell Biol.* **122**, 277–291 (2004).

50. Y. Groemping, K. Rittinger, Activation and assembly of the NADPH oxidase: a structural perspective. *Biochem. J* **386**, 401–416 (2005).

51. X. Liu, *et al.*, Regulation of mitochondrial biogenesis in erythropoiesis by mTORC1-mediated protein translation. *Nat. Cell Biol.* **19**, 626–638 (2017).

52. M. N. Szentirmay, SURVEY AND SUMMARY: Spatial organization of RNA polymerase II transcription in the nucleus. *Nucleic Acids Research* **28**, 2019–2025 (2000).

53. E. J. Wherry, E. John Wherry, T cell exhaustion. *Nature Immunology* **12**, 492–499 (2011).

54. I. Hecht, *et al.*, ILDR2 Is a Novel B7-like Protein That Negatively Regulates T Cell Responses. *J. Immunol.* **200**, 2025–2037 (2018).

55. A. H. Long, *et al.*, 4-1BB costimulation ameliorates T cell exhaustion induced by tonic signaling of chimeric antigen receptors. *Nat. Med.* **21**, 581–590 (2015).

56. S. Krishna, *et al.*, Chronic activation of the kinase IKKβ impairs T cell function and survival. *J. Immunol.* **189**, 1209–1219 (2012).

57. M. Peled, *et al.*, EF Hand Domain Family Member D2 Is Required for T Cell Cytotoxicity. *J. Immunol.* **201**, 2824–2831 (2018).

58. D. Lando, *et al.*, FIH-1 is an asparaginyl hydroxylase enzyme that regulates the transcriptional activity of hypoxia-inducible factor. *Genes Dev.* **16**, 1466–1471 (2002).

59. J.-W. Kim, I. Tchernyshyov, G. L. Semenza, C. V. Dang, HIF-1-mediated expression of pyruvate dehydrogenase kinase: a metabolic switch required for cellular adaptation to hypoxia. *Cell Metab.* **3**, 177–185 (2006).

60. I. Papandreou, R. A. Cairns, L. Fontana, A. L. Lim, N. C. Denko, HIF-1 mediates adaptation to hypoxia by actively downregulating mitochondrial oxygen consumption. *Cell Metab.* **3**, 187–197 (2006).

61. A. L. Doedens, *et al.*, Hypoxia-inducible factors enhance the effector responses of CD8(+)

T cells to persistent antigen. *Nat. Immunol.* **14**, 1173–1182 (2013).

3 Manuscript 2

DeepCLIP: Predicting the effect of mutations on protein-RNA binding with Deep Learning

DeepCLIP: Predicting the effect of mutations on protein-RNA binding with Deep Learning

Alexander Gulliver Bjørnholt Grønning[1,2,3†], Thomas Koed Doktor[1,2†*], Simon Jonas Larsen[3], Ulrika Simone Spangsberg Petersen[1,2], Lise Lolle Holm[1,2], Gitte Hoffmann Bruun[1,2], Michael Birkerod Hansen[1,2], Anne-Mette Hartung[1,2], Jan Baumbach[3,4], Brage Storstein Andresen[1,2]

1 Department of Biochemistry and Molecular Biology, University of Southern Denmark, 5230 Odense M, Denmark.

2 Villum Center for Bioanalytical Sciences, University of Southern Denmark, 5230 Odense M, Denmark.

3 Department of Mathematics and Computer Science, University of Southern Denmark, 5230 Odense M, Denmark.Chair 4 of Experimental Bioinformatics, TUM School of Life Sciences Weihenstephan, Technical University of Munich, 85354 Freising, Germany

† These authors contributed equally.

ABSTRACT

Nucleotide variants can cause functional changes by altering protein-RNA binding in various ways that are not easy to predict. This can affect processes such as splicing, nuclear shuttling, and stability of the transcript. Therefore, correct modelling of protein-RNA binding is critical when predicting the effects of sequence variations. Many RNA-binding proteins recognize a diverse set of motifs and binding is typically also dependent on the genomic context, making this task particularly challenging. Here, we present DeepCLIP, the first method for context-aware modeling and predicting protein binding to RNA nucleic acids using exclusively sequence data as input. We show that DeepCLIP outperforms existing methods for modelling RNA-protein binding. Importantly, we demonstrate that DeepCLIP predictions correlate with the functional outcomes of nucleotide variants in independent wet lab experiments. Furthermore, we show how DeepCLIP binding profiles can be used in the design of therapeutically relevant antisense oligonucleotides, and to uncover possible position-dependent regulation in a tissue-specific manner. DeepCLIP is freely available as a stand-alone application and as a webtool at http://deepclip.compbio.sdu.dk.

INTRODUCTION

The massive technological progress in next generation sequencing (NGS) technologies has made sequencing affordable in the context of precision medicine and personalized health care. NGS analysis enables identification of millions of sequence variants in each patient sample, increasing the need for *in silico* prediction of the functional consequences of a diverse range of variations. In particular, the effect of deep intronic sequence variants at the mRNA level through altered binding to RNA-binding proteins (RBPs) is difficult to predict *in silico* as existing tools' predictions of functional outcomes of splicing are primarily based on the analysis of point mutations within or near exons (1-3). While some existing binding site prediction tools can work on sequences of any type, there is an unmet need for improved modelling of contextual dependencies other than structure that are important for correctly estimating the *in vivo* functionality of the binding sites. Extracted contextual information may form the basis for design of antisense oligonucleotide based therapies, which modulate RBP activity, such as splice-switching oligonucleotides (SSOs) (4-6). Thus, improving information on whether contexts act positively or negatively with regard to binding is an important area of research that will ultimately enable the development of novel therapeutic options in personalized medicine.

Sequencing technologies have also vastly expanded the wealth of information concerning protein binding to RNA when combined with cross-linking and immunoprecipitation (CLIP) techniques (7-9), which allow accurate mapping of protein binding sites in functional *in vivo* contexts. Classically, binding preferences or binding motifs have been represented by position frequency matrices (PFMs). Well-known *de novo* motif discovery tools such as MEME (10) and HOMER (11) output PFMs and base their motif detection and identification on the PFM concept. This approach to motif discovery implicitly assumes that such fixed-length motifs exist and that they function in a context-independent manner regarding the surrounding sequences. They further assume pairwise independence of the nucleotide frequencies within the motifs. However, proteins that bind RNA typically do so in a context dependent manner. In particular, secondary structure may influence the binding of some RBPs (12). Information about double-stranded or single-stranded structure has been incorporated into MEMERIS

(13), which is an extension of the MEME algorithm. Further structural dependencies have been incorporated into RNAcontext (12), which expands the information about secondary structure from simple double or single-stranded structures into paired, hairpin loops, bulges and internal or multi-loops, and unstructured contexts in order to further optimize the modeling of binding preference of RBPs. More recently, a graph-based modeling of structural and sequence binding preferences was introduced in the GraphProt (14) software, which out-performed RNAcontext on a set of diverse CLIP datasets using different CLIP methods. GraphProt uses RNAshapes (15) to predict the structures of RNA sequences, which are then encoded into a hypergraph from which important structural features can be extracted. To improve the structure estimations, GraphProt extends the CLIP-derived sequences by 150 bp in each direction. Together with sequence features extracted only from the CLIP-derived binding sites, an overall model of binding preference is generated using support vector machines.

While inclusion of structural preferences may increase accuracy in prediction, these models still fail to capture other contextual dependencies affecting the *in vivo* functionality, such as a high density of protein binding sites nearby or localization within a specific functional region of the transcript, such as proximity to splice sites. For instance, exonic splicing enhancers (ESEs) that enhance splicing of exons by binding to SR proteins are enriched in exons, while exonic splicing silencers (ESSs) are underrepresented in exons. These observations have been used to generate ESE and ESS motifs from sequences enriched (16,17) or depleted (16) in exons. Such contextual dependencies were recently introduced in the iONMF software(18), which uses integrative orthogonality-regularized nonnegative matrix factorization to incorporate multimodal information about CLIP-derived binding sites such as their position within the gene (5'UTR, CDS, exon, intron, 3'UTR), gene ontology, and presence of other protein binding sites determined by other CLIP studies, in addition to structural information, which improved performance for some datasets.

In recent years, deep learning techniques have been used to model protein binding. Deep learning has proven successful in various difficult classification tasks such as natural language processing (19), object recognition (20) and reconstructing brain circuits (21). Deep learning allows computational models composed of multiple processing layers to learn representations of data with multiple levels of abstraction (22). Deep learning models can identify dependencies and complex structures in very high-dimensional data - such as CLIP data - and have been used, for example, for predicting the effects of mutations in non-coding DNA on gene expression and disease (1,23), predicting DNA function (24), mRNA coding potential (25), and prediction of subcellular locations of proteins (26). Starting with DeepBind (27), which is trained on *in vitro* RNAcompete data, convolutional neural networks (CNN) have been used to estimate binding affinity, and later methods such as DLPRB (28) have expanded on this to also use secondary structure as well as recurrent neural networks trained on *in vitro* RNAcompete data.

Using *in vivo* CLIP training data, Deepnet, a multimodal deep belief network incorporating 2D structure information (mDBN-) or both secondary and tertiary structure information (mDBN+) along with a CNN architecture was introduced in the deepnet-rbp software (29), while more recently the iDeep framework combines the annotation data used by iONMF with a CNN into a multimodal neural network with increased accuracy in classification compared to iONMF(30). Even more recently, iDeepS was introduced as a replacement for iDeep to include analysis of 2D structural motifs much like GraphProt, using a combination of CNN and bidirectional LSTM (Long Short-Term Memory)

layers akin to DeepCLIP's architecture (31). Previous models for RBP binding properties that consider contextual clues are focused either specifically on structural dependencies, which may fail to capture other important contextual dependencies, or on the presence of annotation data to aid in the task of classification. However, static annotations will not contribute to determining the effect of a mutation on the binding activity of proteins. For instance, a model, which relies heavily on clues from annotation data about the genomic region, such as location within an exon, will be unable to use this level of information to ascertain the effect of an exonic point mutation in which the context is maintained. Only iDeepS has a general-purpose LSTM layer able to model general context dependency, but it is supplemented with structural predictions from an external program and thus does not work on sequence data alone. Understanding binding preferences is important for evaluation of the phenotypic impact of sequence variations. Mutations may alter the phenotype at several different levels, as in the case of missense mutations, which in addition to altering the amino-acid sequence, may also change the splicing pattern (32). Other mutations with less *visible* deleterious effects may abolish healthy splicing by altering the binding of RBPs, sometimes at somewhat distant sites. Splicing and the overall binding activity of RBPs is the result of a balance between positively and negatively acting elements that cooperate or compete for binding (33),(34), so even minor changes in RBP binding sites can change the outcome of splicing events.

Importantly, before they can be applied in predicting clinically important changes or functional elements to be targeted, binding models needs to be validated in the laboratory, using *in vitro* techniques such as RNA-protein affinity measurements and *in vivo* techniques such as minigene transfections and predictions need to be consistent with effects reported in clinically affected patients.

In this paper, we present DeepCLIP, a novel deep learning based tool for discovering protein-RNA binding sites and for characterizing binding preferences of RBPs. We demonstrate how it outperforms current state-of-the-art RBP binding analysis tools, and we show that DeepCLIP's predictions provide information about high-affinity RBP binding sites and that it successfully predicts alterations of the RBP affinity for RNA sequences when single nucleotide polymorphisms (SNP) or disease-causing mutations are introduced. This is reflected both in the binding profiles that show the region(s) important for RBP binding, and in the predictions of the sequences which indicate whether they are more similar to the "consensus" CLIP-sequence or to the genomic background. Last but not least, we have made DeepCLIP available as an online tool for training and application of protein-RNA binding deep learning models and prediction of the potential effects of clinically detected sequence variations (http://deepclip.compbio.sdu.dk/). We also provide DeepCLIP as a configurable stand-alone program (http://www.github.com/deepclip).

MATERIALS AND METHODS

DeepCLIP: more than just a motif discoverer

DeepCLIP is essentially a deep neural network that uses shallow 1D convolutional layers to find and enhance features of a set of presented sequences (26,35,36). This is followed by a Bidirectional Long Short Term Memory (BLSTM) layer (37,38) which uses the extracted features and contextual information of the sequences to find areas of the RNA-sequences associated with RBP binding (Figure

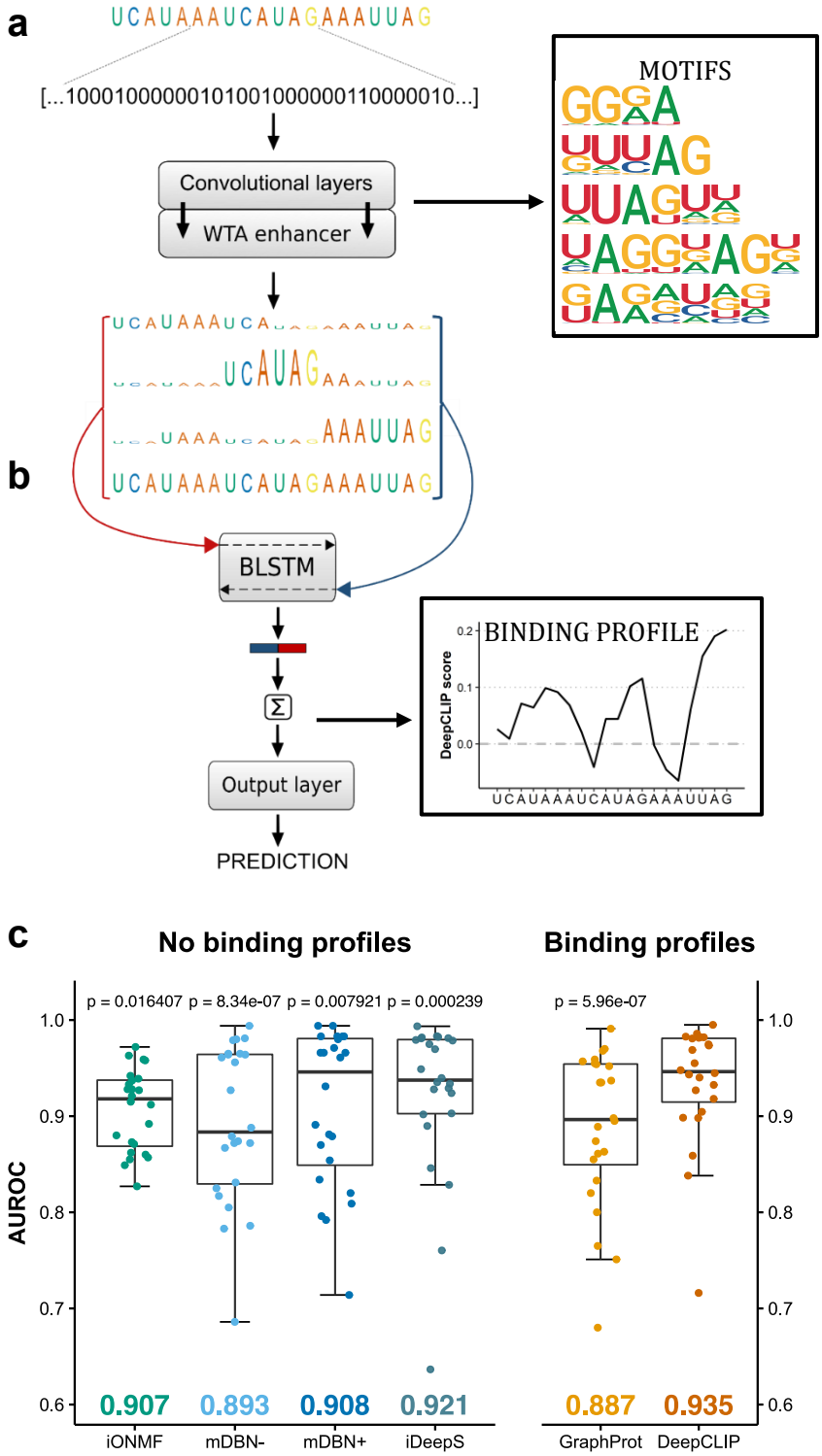


Figure 1 | Classification performance of DeepCLIP surpasses competing methods. (a) One-hot embedded RNA-sequences function as input to the neural network and the 1D convolutional layers, which are further enhanced by a winner-takes-all (WTA) layer. (b) The outputs are concatenated with the input sequence and each segment of the concatenated arrays that corresponds to aligned bases is introduced to the following BLSTM layer as individual time-steps. The WTA-enhanced RNA-sequences are the three uppermost and the input sequence is in the bottom. DeepCLIP produces a prediction score, which is used during training, as well as binding motifs and a binding profile. (c) Boxplot of comparative analyses of DeepCLIP classification performance against other state-of-the-art tools. Area under receiver operator characteristic curve (AUROC) were measured in 10-fold cross-validation and statistical significance was computed through an Wilcoxon signed-rank test. P-values above individual tools are from pair-wise comparisons with DeepCLIP. Mean AUROC score is indicated in the bottom.

1). Initially, the convolutional layers of DeepCLIP can be regarded as a collection of randomly generated PFMs of user-defined sizes that, as training progresses, learn to recognize important nucleotide patterns in the input data. When predicting, the convolutional layers score sequence segments according to their importance for the classification task. Pseudo-PFMs can be generated by collecting scored sequence patterns and counting the frequencies of different nucleotides at each possible position. We use the term pseudo-PFMs because each sequence used for the PFM generation is weighted by the squared output score given by the convolutional layers. In this way, the pseudo-PFMs will depict important class-specific nucleotide patterns. The BLSTM layer of DeepCLIP is used to generate a binding profile at the nucleotide level. The BLSTM layer consist of two LSTM layers that analyze "hidden" sequence representations (modified outputs of the convolutional layers) in a bidirectional manner (Figure S1).

The DeepCLIP tool takes a single or more RNA oligonucleotides (short RNA sequences) as input and predicts binding probability and calculates a binding profile. The main purpose of DeepCLIP is to identify binding sites of proteins in novel untested sequences using trained models that have extracted binding site information provided by CLIP data, to predict the effect of sequence variants on the binding, and to identify the importance of individual nucleotides for protein binding affinity. DeepCLIP can be run on a standard PC with Linux installed as operating system. Training of smaller datasets require at least 4GB RAM, but larger datasets will require more memory. The largest dataset used for training in this study consisted of a total of 1,256,372 input sequences, with a memory footprint of app. 43GB RAM. DeepCLIP is fast enough to run online on a web server (http://deepclip.compbio.sdu.dk/) and its Python code is also publicly available (http://www.github.com/deepclip).

Training workflow of DeepCLIP

The core of DeepCLIP is a convolutional BLSTM network implemented in Theano (39) using the Lasagne library (40) and a few customized network layers and functions. DeepCLIP is a binary classifier that uses supervised learning to distinguish between unbound sequences and bound sequences derived from CLIP-experiments. The input to DeepCLIP consists of positive (bound) sequences, which are assigned to class 1, and negative (unbound) sequences assigned to class 0. These input sequences are converted into linearized one-hot encoded vectors, which serve as the actual input to the neural network layers. The DeepCLIP architecture is shown in Figure 1.

By default, 80% of the input data is used for training while 10% is used for validation and the last 10% for testing (Figure S1d). DeepCLIP is trained by iterating over the training and validation sets several times (also called epochs). While training, performance is measured on the validation set after every epoch and the best performing model is saved. Early stopping can be applied to prevent training after a likely maximum performance has been obtained. The final performance of the saved model is measured on the test set, which contain data that have not previously been introduced to the model. When running in 10-fold cross-validation the input sequences are divided into 10 equal sized bins, and each bin is used once as a test set, once as a validation set and 8 times as part of the training set (Figure S1e).

Encoding of sequence data

DeepCLIP processes sequence data as linearized one-hot encoded vectors. In the one-hot representation, the items of the vocabulary, *v = (A, C, G, U)*, are represented by vectors with lengths equal to the length of the vocabulary that each have a 1 in unique dimensions. The one-hot encoded bases are therefore independent of one another and are equally similar or dis-similar. It signals that no prior correlations between the bases are known. In this way, the network will determine correlations between the bases on its own (41). All input sequences are zero-padded until they have identical lengths and until the largest filter can conduct "full convolutions" as it is defined in the Lasagne documentation (40). Following vectorization of the bases of the sequences, the combined vectors are linearized, and the resulting one-dimensional data used as input to the neural network.

Convolutional neural network layer

Convolutional layers consist of nodes that are only sensitive to a defined receptive field referred to as kernels or filters. Nodes of convolutional layers apply weight-sharing and sparse-connectivity, which means that the same filter can "view" all possible filter-sized segments of the input individually (42).

As in previous work (26,27,29,30), the filters of the convolutional layers in DeepCLIP can be interpreted as motif detectors. The sizes of the filters of the convolutional layers are optional but we used ranges from 4-8 one-hot encoded bases. DeepCLIP only applies a single filter of each size. The strides of the filters are $|v|$, so the filters only convolve patterns consisting of whole one-hot encoded bases. The convolutional layers apply the rectifier activation function (43). In this context, it means that only patterns that receive a score above zero can be assumed important for sequence identification.

The bias parameters of the convolutional layers are removed to ensure that only areas in the sequence input containing one-hot encoded bases produce outputs above 0, preventing recognition of zero-padded areas alone. As the nodes in the convolutional layers are rectifying linear units, only sequence-segments that are deemed important by the neural network will produce convolutional-outputs above 0. The initial weights of the convolutional nodes are set to 0.01. The filter sizes allow for diverse sequence patterns of various length to be incorporated into the model. The output vectors of convolutional layers in DeepCLIP are vectors containing values between 0 and ∞. Before the output vectors are passed to the BLSTM layer they undergo a so-called WTA-enhancement (Winner Take All-enhancement), which is described below.

The single highest values of the different output vectors (44) are multiplied by 2 which is followed by a squaring of the vectors to enhance differences between high and low values. These squared vectors have now been WTA-enhanced, where the "winners" are the highest values in the output vectors. The WTA-enhanced vectors are used for a recreation of the original one-hot embedded sequences where the one-hot values are defined by the WTA-enhanced vector elements. For each convolutional layer, a WTA-enhanced sequence is created and concatenated in a manner that makes it possible to process each numerical base representation of the sequences as individual steps in the following BLSTM layer (see Figure 1).

In this way, the convolutional layers help guide the attention of the BLSTM layer. The pseudo-PFMs created by the DeepCLIP tool that depict the important patterns in the RBP-bound sequences, are

based on the specific sequential areas that only relate to class 1. Meaning, if an area of a sequence is, by the BLSTM layer, predicted as being associated with class 0, any convolutional output values in the sequential area will be zeroed out and therefore will not be a part of the pseudo-PFM calculation. The filters of the convolutional layers of the model with the best performance in the 10-fold cross validation were extracted and used for creation of pseudo-PFMs. The top 1000 sequences with respect to the predictions were used for the pseudo-PFM calculation.

Bidirectional LSTM layer

Long short-term memory (LSTM) networks have already proven successful in biological sequence analysis (26,45). DeepCLIP uses a single BLSTM layer that processes the WTA-enhanced sequences (Figure 1). In BLSTM layers, the input sequences are presented forwards and backwards in two separate LSTM layers that are connected to the same output layer (38). The implementation of a single LSTM layer in DeepCLIP is given by equations (1-9):

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_{xi}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \quad (1)$$

$$\boldsymbol{f}_t = \sigma\big(\boldsymbol{W}_{xf}\boldsymbol{x}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f\big) \quad (2)$$

$$\boldsymbol{g}_t = \tanh\big(\boldsymbol{W}_{xg}\boldsymbol{x}_t + \boldsymbol{W}_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g\big) \quad (3)$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t \quad (4)$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_{xo}\boldsymbol{x}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \quad (5)$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \tanh(\boldsymbol{c}_t) \quad (6)$$

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (7)$$

$$\tanh(z) = \frac{e^{2z} - 1}{e^{2z} + 1} \quad (8)$$

$$\text{Hadamard product} = \odot \quad (9)$$

where $\boldsymbol{i}, \boldsymbol{f}, \boldsymbol{g}, \boldsymbol{o}$ and $\boldsymbol{c}$ are the input gate, forget gate, modulatory gate, output gate and cell, respectively. $\boldsymbol{x}_t$ is the input vector at timestep t, $\boldsymbol{W}_{xi}$ is the input-input gate weight matrix, $\boldsymbol{W}_{hi}$ is the hidden-input gate weight matrix, $\boldsymbol{b}_i$ is the bias of the input gate and $\boldsymbol{h}_{t-1}$ is the hidden output vector from timestep $t-1$. The same logic applies for the remaining gates. σ is the sigmoid activation function, $tanh$ is the hyperbolic tangent activation function and the Hadamard product indicates elementwise multiplication. The hidden output vector of a LSTM memory block is $\boldsymbol{h}_t$ at timestep t.

The forward LSTM reads an input sequence with length T from $\boldsymbol{x}_1$ to $\boldsymbol{x}_T$ and the backward LSTM reads the same input sequence from $\boldsymbol{x}_T$ to $\boldsymbol{x}_1$. The forward LSTM layer produces forward hidden vectors, $\overrightarrow{\boldsymbol{h}_1}, \overrightarrow{\boldsymbol{h}_2} \dots \overrightarrow{\boldsymbol{h}_T}$ and the backward LSTM layer produces backward hidden vectors $\overleftarrow{\boldsymbol{h}_1}, \overleftarrow{\boldsymbol{h}_2} \dots \overleftarrow{\boldsymbol{h}_T}$. Here, the output of the backwards LSTM layer has been reversed, so the outputs of forward and backward LSTM layers go from $\boldsymbol{x}_1$ to $\boldsymbol{x}_T$. The hidden vector of the BLSTM layer at time step t, $\boldsymbol{h}_t$,

is given by the concatenation of the forward hidden vector and the backward hidden vector $\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}_t}; \overleftarrow{\boldsymbol{h}_t}]$ (46). The output sequence of a BLSTM layer given an input sequences of length T can be seen as a matrix, $\boldsymbol{H} = (\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots \boldsymbol{h}_T)$, where each $\boldsymbol{h}_t$ is a row. Each $\boldsymbol{h}_t$ contains information about the whole input sequence with a strong focus on the parts surrounding the t^{th} input vector (46). In DeepCLIP, each $\boldsymbol{h}_t$ represents a base of a sequence that has knowledge of the surrounding sequence. These context-aware representations of sequences function as input to the output layer where the final prediction is calculated. Dropout is applied on $\boldsymbol{H}$, which means that recurrent connections are not affected.

Output layer, binding profile and prediction

The output layer consists of a single fixed node without a bias parameter that uses the sigmoid activation function. By "fixed" we mean that the parameters of the node do not update when training. The initial weight of the node is set to 1.0, which means that the node is forced to associate positive values with class 1 and negative values with class 0. The input to the output layer, given a single input sequence, is the vector that results from a row summation of the matrix $\boldsymbol{H}$ where segments based on zero-paddings are zeroed out. By zeroing out these segments it is ensured that only areas that contain one-hot embedded bases are used for the prediction of the given sequence. Basically, the prediction of a given RNA-sequence is the sum of the output values of the BLSTM layer inserted into a sigmoid activation function.

In terms of the BLSTM output, if a $\boldsymbol{h}_t$ is mainly positive the base at its specific position is associated with the "consensus patterns" of the sequences derived from the CLIP experiments. If a $\boldsymbol{h}_t$ is primarily negative, the base at its specific position is more associated with random background sequences derived from the genome. And if the values of a $\boldsymbol{h}_t$ sums to ~0, the base at its position could belong to both classes. This approach makes DeepCLIP able to highlight sequential areas that differ from genomic background and thereby able to identify *in vivo* binding sites. The binding profiles are constructed using the input vectors of the output layer, where all the zero-padding has been removed.

DeepCLIP default settings

DeepCLIP uses ADAM(47) ($\alpha = 0.0002, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$) for gradient descent optimization. For the BLSTM layer, the parameters are sampled from a Gaussian distribution with $\mu = 0.0$ and $std = 0.01$. DeepCLIP uses binary cross entropy as loss function and employs dropout in order to avoid overfitting. The most optimal network weights based on Area Under Receiver Operator Curve (AUROC) performance on the validation set are saved during training. Dropout is applied to the BLSTM layer (10%).

Generation of background sequences

Background sequences can be either supplied by the user as sequences or generated automatically by DeepCLIP from positive binding sites in one of two ways. The default way is to supply positive binding site in BED format and then generate a random set of identically sized genomic regions as the positive binding site and randomly placing each within the same gene as the corresponding positive

binding site such that no background regions overlap either a positive binding site or another background region. Alternatively, background sequences can be generated from positive binding sites by scrambling the input sequences. When positive binding sites are supplied in BED format, they can optionally be expanded on each size, or fixed to a certain width, or a combination of these. In this study, we have used the random genomic background method to most accurately obtain *in vivo* non-bound sites.

Analysis of Area Under Receiver Operator Curve classification performance

Comparison of Area Under Receiver Operator Curve (AUROC) classification performance was performed on a dataset compiled in a previous study paper (14). In order to minimize computational complexity, we took the performance numbers of alternative models on this dataset as they were reported in previous studies (14,18,29). We could not compare to iDeep (30), as they did not provide numbers for the same datasets, and did not provide any way of producing the multimodal data required as input. We ran iDeepS on 101 nt sequences from the GraphProt dataset by expanding the peak-areas on either side until the sequence was 101 nt. We ran iDeepS on these sequences with the same number of epochs that we used to train DeepCLIP models. We generated 10-fold cross-validation sets where one set was held out for testing one time, and used in training the other 9 times, in order to obtain comparable performance measures across the full datasets. We ran DeepCLIP on the peak-area sequences of the datasets in a 10-fold cross-validation, such that each site was held out exactly once for validation during training, and once for final testing, while being used for actual training the remaining eight times. Model performance was measured for each dataset using the performance measure tool "perf" as used by GraphProt on the combined predictions from all CV cycles. Additionally, AUROC confidence intervals were estimated using the DeLong algorithm as implemented in the "pROC" R package (48).

Benchmark analysis on eCLIP data

Binding sites for hg38 from ENCODE (49) were downloaded and replicate experiments were combined into the overlapping regions of both replicates to obtain high-confidence binding sites from which we extracted sites with lengths 12-75 nt plus 150 nt padding to enable GraphProt compatibility. A balanced negative background set was constructed using DeepCLIP's implementation, namely shifting each positive binding site to a random site within the same gene that does not overlap positive binding sites or other previously allocated negative binding sites. Similarly, we downloaded eCLIP binding sites from the POSTAR2 database (50), merged all sites to remove any duplicates and extracted sites in the range 12-75 nt with additional 150nt padding, which we used to construct a balanced background set. Sequences were trimmed to obtain an equal length of 101 nt for use with iDeepS, and padding removed to obtain sequences of 12-75 nt to use with DeepCLIP models. After creating these balanced datasets we used either the published models (GraphProt) or models we had trained ourselves (iDeepS and DeepCLIP) to measure classification performance using the AUROC metric.

Additional models from public CLIP data

Binding sites from an eCLIP study of SRSF1 (49) in K562 cells were downloaded and the overlap between two replicates were extracted to create a set of non-redundant binding sites. These were used for constructing the bound dataset, with a matched genomic background as control. We then trained an SRSF1 DeepCLIP model on this dataset using same running parameters as previous models, with 50 training epochs and early stopping after 5 epochs. TDP-43 binding sites were downloaded from POSTAR2 (50) and non-redundant input sites were used to train a DeepCLIP model, again using identical running parameters as previous models, but adjusting the number of training epochs to 50 and early stopping after 5 epochs to account for the much larger training set. Similarly, hnRNP A1 binding sites (51), were used to train a DeepCLIP model on binding sites with p-values below 0.01 using default parameters with 200 training epochs and early stopping after 20 epochs. In all cases, 10-fold cross-validation was used to identify the best performing model.

Additional RNAcompete based models

We trained binary classification models from RNAcompete data (52) by sorting the sequences by the normalized intensities and setting the 1,000 highest scoring as the positive class, and the 1,000 lowest scoring as the negative class. Sequences were zero-padded to allow models trained on the shorter RNAcompete sequences to predict class of the 12-75 nt GraphProt benchmark datasets. We then used the models to obtain AUROC estimates for the corresponding GraphProt dataset.

Analysis of NM_032776.1 transcript with PUM2 and QKI models

DeepCLIP was run in long-prediction mode with the PUM2 and QKI models trained on the GP dataset to produce binding profiles across the length of the transcript (8762 nt total). A sliding window of 9 nt was then used to identify regions with a minimum mean profile score above 0.3. Overlapping 9 nt windows were combined to produce a non-redundant set of predicted binding sites. Mapping locations relative to the transcript of predicted as well as observed binding sites from eCLIP (49) and PAR-CLIP (8) were visualized with Gviz (53).

Analysis of TDP-43 repressed pseudoexons

Pseudoexons activated by conditional knock-out of TDP-43 in mice (54) were analyzed with DeepCLIP by first extracting the sequence of the pseudoexon along with 100 nt of the neighboring intronic sequences. These sequences were then used to produce binding profiles by using a sliding window approach to produce raw DeepCLIP profiles of smaller segments, taking the value of the central nucleotide to build a binding profile covering the entire length of the sequence. Subsequently, regions corresponding to the 25 first and last nucleotides of the exons along with the 50 first and last nucleotides of the neighboring introns were extract in order to analyze TDP-43 binding to the acceptor and donor splice site regions.

Minigene generation

ACADM exon 5 minigenes were identical to the previously used wt *ACADM* minigene (34), with the exception of nucleotide variants at positions corresponding to c.361, c.362, and c.363 with exon 5. These variants were introduced as previously described (34).

The *ACADM* exon 6 wt minigene was generated from genomic DNA by amplifying the complete exon 6 (81 bp) along with 864 bp of intron 5 and 603 bp of intron 6 and subsequent cloning into the pSPL3 vector (Gibco BRL) using the BamHI and XhoI restriction sites. For amplification we used the forward primer 5'-TCGAGAATTCAGGAGCA-3' and the reverse primer 5'-CTCCACTAAATAGAGC-3'. The IVS6+7A>G mutation was introduced by GenScript (GenScript, Piscataway, NJ, USA).

***ACADM* exon 5 minigene transfections and RT-PCR.**

HEK-293 cells were seeded in 3.5 cm^2 12-well plates (Nunc) at a density of $4x10^5$ cells/well 24 hours prior to transfection. In each well, cells were transiently transfected using X-tremeGENE 9 DNA Transfection Reagent (Merck): 0.3 µg of one of the *ACADM* exon 5 minigenes c.362C (wildtype), c.361C, c.361G, c.361T, c.362A, c.362G, c.362T, c.363A, c.363C, or c.363G. After 48 hours of incubation following minigene transfection, cells were harvested using QIAzol Lysis Reagent (Qiagen), followed by phenol/chloroform extraction of total RNA. Reverse transcription was performed using the High Capacity cDNA Reverse Transcription Kit (Thermo Scientific). Splicing patterns were analyzed by PCR amplification, using TEMPase Hot Start DNA Polymerase (Ampliqon), and agarose gel electrophoresis. We used the *ACADM* exon 5 minigene specific primers: MCTEST2AS (5′-AGACTCGAGTTACTATTAATTACACATC-3′) and MC242S (5'-CCTGGAACTTGGTTTAATG-3'). PCR products were quantified according to molar ratios by capillary gel electrophoresis on a Fragment Analyzer™ instrument (Agilent), and visualized on 1.5% agarose gels. Experiments were performed in triplicates.

***ACADM* exon 6 minigene transfections with siRNA mediated knock-down of TDP-43 and RT-PCR.**

Knockdown of TDP-43 was obtained by performing reverse transfection during initial seeding of cells and another transfection 48 hours later. Both transfections were performed using Lipofectamine RNAiMAX Transfection Reagent (Thermo Fisher Scientific) and 40 nM of siRNA targeting *TARDBP* (L-012394-00-0020, Dharmacon) or non-targeting siRNA (D-001810-10-20, Dharmacon). HeLa cells were seeded in 3.5 cm^2 12-well plates (Nunc) at a density of $1.5x10^5$ cells/well 24 hours prior to minigene transfection. In each well, cells were transiently transfected using X-tremeGENE 9 DNA Transfection Reagent (Merck): 0.4 µg of one the two *ACADM* exon 6 minigenes: WT or +7A>G. After 48 hours of incubation following minigene transfection, cells were harvested using QIAzol Lysis Reagent (Qiagen), followed by phenol/chloroform extraction of total RNA. Reverse transcription was performed using the High Capacity cDNA Reverse Transcription Kit (Thermo Scientific). Splicing patterns were analyzed by PCR amplification, using TEMPase Hot Start DNA Polymerase (Ampliqon), and agarose gel electrophoresis. We used the minigene specific primers: SD6 (5'-TCTGAGTCACCTGGACAACC-3') and SA2 (5'-ATCTCAGTGGTATTTGTGAGC-3'). PCR

products were quantified according to molar ratios by capillary gel electrophoresis on a Fragment Analyzer™ instrument (Agilent), and visualized on 1.5% agarose gels. Knockdown of TDP-43 was validated by SDS-PAGE and Western Blotting and membranes were probed with antibodies anti-TDP-43 (10782-2-AP, ProteinTech) and as a loading control anti-HPRT (HPA006360, Merck). Experiments were performed in triplicates.

SSO co-transfection with *ACADM* exon 6 minigenes

HeLa cells were reverse transfected in duplicates with 40 nM SSO using Lipofectamine RNAiMAX Transfection Reagent (Thermo Fisher Scientific) according to the manufacturer's protocol, and seeded in 3.5 cm^2 12-well plates (Nunc) at a density of $2x10^5$ cells/well 24 hours prior to minigene transfection. SSOs were phosphorothioate oligonucleotides with 2′-O-methyl modifications on each sugar moiety (LGC Biosearch Technologies): SSO1 (5'-UAAGUGUGAAAUAAAGCGGCAGUUA-3'), SSO2 (5'-AGUGUGAAAUAAAGCGGCAGUUACA-3'), or a control SSO without any human target sites: 5'-GCUCAAUAUGCUACUGCCAUGCUUG-3'. Cells were transiently transfected using X-tremeGENE 9 DNA Transfection Reagent (Merck): 0.4 µg of one of the two *ACADM* exon 6 minigenes: WT, or +7A>G. After 24 hours of incubation following minigene transfection, cells were harvested using QIAzol Lysis Reagent (Qiagen), followed by phenol/chloroform extraction of total RNA. Reverse transcription was performed using the High Capacity cDNA Reverse Transcription Kit (Thermo Scientific). Splicing patterns were analyzed by PCR amplification, using TEMPase Hot Start DNA Polymerase (Ampliqon), and agarose gel electrophoresis. We used the minigene specific primers: SD6 (5'-TCTGAGTCACCTGGACAACC-3') and SA2 (5'-ATCTCAGTGGTATTTGTGAGC-3'). PCR products were quantified according to molar ratios by capillary gel electrophoresis on a Fragment Analyzer™ instrument (Agilent), and visualized on 1.5% agarose gels. Experiments were performed in triplicates.

Surface plasmon resonance imaging method

Biotinylated oligonucleotides were immobilized on a Senseye G strep (SSENS) sensorchip in a 2x4x12 array by continuous flow in a CFM 2.0 printer (Wasatch microfluidics). The oligonucleotides were diluted in 1XTBS to a concentration of 1 µM and spotted for 20 min followed by 5 minutes washing with TBS + 0.05% Tween-20. The sensor chip was transferred to the MX-96 (IBIS technologies) , and the system was primed with SPR buffer (10 mM Hepes/KOH pH 7.9, 150 mM KCl, 10 mM $MgCl_2$ and 0.075% Tween-80). Surface plasmon resonance imaging (SPRi) by IBIS MX-96 was used to measure the kinetics of recombinant hnRNP A1 (ab224866, Abcam), SRSF1 (GenScript, Piscataway, NJ, USA) and TDP-43 (R&Dsystems, AP-190) binding to the immobilized RNA oligonucleotides. Binding was measured in real time by following changes of the SPR angles at all printed positions of the array during 10 min. injections of recombinant protein over the entire surface. Seven injections of a 2-fold titration series from 6.25 to 400 nM protein was injected in sequence from the lowest concentration to the highest. Before adding protein to the chip, residual background binding was blocked by injecting 20mg/ml BSA in SPR buffer onto the chip for 10 minutes. A continuous flow of SPR buffer flowed over the surface before, between and after the protein injections, to measure baseline and dissociation kinetics. Dissociation was measured for 8

minutes, by injecting SPR buffer over the chip at a rate of 4 µl/sec. Responses for a calibration curve were created after the concentration series by measuring SPR responses from defined dilutions of glycerol in running buffer (ranging from 5 to 0 % glycerol) and of pure water as defined by the automated calibration routine of IBIS MX-96.

Data analysis: The SPRi data was imported into SPRINTX software (v. 2.1.1.0, IBIS technologies), calibrated, reference subtracted, and the baseline of the responses before all injections were zeroed. The time starting point was aligned at the beginning of each new injection. Then the data were exported to Scrubber 2 (v 2.0c, Biologics Inc.). Binding curves for all chip positions where binding was observed were fitted globally to the integrated rate equation that describes simple first order 1:1 binding kinetics to obtain kinetic association rate (ka), dissociation rate (kd) as well as the Rmax for the binding model. For hnRNPA1 and TDP-43 a 1:2 biphasic model was calculated and fitted. ClampXP (version 3.50, Biosensor Data Analysis) was used with a bimodal model to fit the binding data. The secondary Ka and Kd parameters were fixed to 1e-5 M, due to very low secondary association and dissociation. Primary binding parameters and ligand concentration (Rmax) were set to float. SPRi measurements were performed twice, with technical duplicates being used for model fitting each time.

Statistical analyses

All statistical analyses were performed in R (version 3.5.3). We used the default wilcox.test() for two-tailed Wilcoxon rank sum and Wilcoxon signed rank tests. Linear regression was carried out using ggplot2 (version 3.2.1) and geom_smooth(method=lm, colour="red", se=TRUE), with correlation significance analysis using the default cor.test() with method="spearman".

RESULTS

DeepCLIP outperforms structural and multimodal models from sequence data alone

DeepCLIP is a neural network that combines shallow convolutional layers with a small bidirectional long short-term memory network to produce both a binding profile and a classification score ranging from 0 to 1 (Figure 1, S1a-c). Models are created by training a network on a set of known binding sites and a set of background genomic sequences (Figure S1d,e), which can optionally be generated by DeepCLIP by providing binding locations instead of raw binding sequences.

To ascertain DeepCLIP's classification performance on a standardized dataset, we generated models from the curated CLIP datasets (8,9,55-63) used in the GraphProt publication (14), which has previously been used in other studies (18,29). First, we trained DeepCLIP models in a 10-fold cross-validation scheme using 50-500 epochs depending on the size of the individual dataset with early stopping after 10% of the maximum number of epochs (Table S1). Next, we measured area under receiver operator characteristic curve (AUROC) using the standard method of 10-fold cross-validation and the combined performance across the 10 different sets (Figure S1e, Table S2). Importantly, DeepCLIP does not directly model structure. Consequently, we used the peak area alone, which is akin to the viewpoint mechanism as adopted in GraphProt. We compared the performance of our models with the performance numbers reported in the earlier studies describing GraphProt, iONMF, and deepnet-rbp (mDBN- and mDBN+). To obtain AUROC values for iDeepS, we performed a 10-

fold cross-validation using the curated CLIP-datasets, as this was omitted in the iDeepS paper, by extending the peak area to 101 nt per the input requirement for iDeepS. We found that DeepCLIP was the overall best classifier in every pair-wise comparison and when looking at the mean AUROC score, underscoring that DeepCLIP performs well on a broad set of data. Furthermore, DeepCLIP had more narrow distributions of scores with fewer low-scoring datasets and a majority of datasets scoring above 0.9 (Figure 1c). DeepCLIP consistently ranked among the best classifiers on individual datasets (Figure S2a, Table S3), even without additional contextual data, i.e. working on the sequence input data alone. In total, DeepCLIP had the best performance for 14 of the 24 datasets, and second best for 6 datasets. For TAF15, mDBN+ and DeepCLIP had identical scores when rounding to the 3rd decimal. DeepCLIP was also a better classifier than DeepBind ((pseudo)median 15.16 %-points (CI-95%: 7.76-24.17), Wilcoxon signed-rank p=0.0004883), on the 12 datasets for which DeepBind models are available (Figure S2b,c).

We chose the best model based on AUROC measure on the validation set, but a more conventional choice is to use the validation loss metric to select the best model. In a post-hoc analysis we also implemented loss-based model selection and found that AUROC performance on the held out test set improved, but only to a small degree (Figure S3). We did not observe any significant differences in DeepCLIP AUROC scores between the CLIP methods (Figure S4a), a significant correlation to the number of bound sites (Figure S4b, $p = 0.647$), or the mean input length of the training data (Figure S4c, $p = 0.118$), but a tendency towards improved performance on the nucleotide-resolution CLIP datasets (iCLIP and PAR-CLIP) versus HITS-CLIP did appear. We did, however, observe a significant negative correlation between GC-content and AUROC performance (Figure S4d, $p = 7.89*10^{-6}$), which seems to be primarily driven by a positive correlation with U-content (Figure S4e, $p = 0.000356$) and a negative correlation with G-content (Figure S3h, $p = 1.04*10^{-5}$). Since many of the proteins in the dataset bind U-rich motifs, and U is known to cross-link more efficiently than other bases, this indicates a general CLIP bias in the models, where the model recognizes not a specific binding site, but a CLIP site in general. In particular, PAR-CLIP is known to display U-bias due to protocol specific treatment of the samples.

We therefore benchmarked the models against completely independent eCLIP datasets from ENCODE (49) and the POSTAR2 database (50). We were not able to run iONMF or mDBN models for these datasets, but compared to GraphProt and iDeepS, DeepCLIP performed favorably on both with both types of model selection (Figure S5a,b, Table S4). In general, there were many models that performed poorly with AUROC values close to or below 0.5. We therefore also filtered out models scoring below 0.6 in all datasets, to focus on just the models with reasonable performance by at least one method. All methods were more similar in this analysis, with no method having a clear advantage over the others. Notably, in other studies using *in vitro* binding data to train models, a better performance was demonstrated on the GraphProt benchmark dataset. We therefore trained DeepCLIP models on corresponding RNAcompete datasets by setting the 1,000 highest scoring RNAcompete sequences as the positive class, and the 1,000 lowest scoring sequences as the negative class. We then compared the performance of these models on the GraphProt dataset to the performance of RCK (64), RNAcontext (12), DeepBind (27), DLPRB-CNN and DLPRB-LSTM (28) (Figure S5c, Table S5). We found that DeepCLIP had the overall highest mean performance with a more narrow distribution of scores, indicating excellent performance of DeepCLIP trained on *in vitro* data compared to existing state-of-the-art methods.DeepCLIP performed well on all CLIP datasets regardless of size and CLIP

method, with the exception of ALKBH5, which is a problematic dataset for all methods that do not rely on additional metadata, presumably due to non-specific binding that may take place in cooperation with a number of other factors that target the factor to specific regions within the transcript. DeepCLIP is thus a robust classifier of *in vivo* binding sites using only sequence data. It compares favorably to models employing external structural information and annotation data in addition to sequence data.

DeepCLIP models predict binding motifs

Although the classifications of DeepCLIP are based entirely on the values of the binding profiles, motifs can be assessed from the CNN filters incorporated in the network architecture. The motif of each filter was generated using the patterns from the 1,000 input sequences that produced the highest DeepCLIP classification score. We found that DeepCLIP produces motifs that are visually similar to previously published motifs (65-70), illustrating that DeepCLIP's classification performance is not simply a result of learning how to recognize the background sequences, but depends on the binding preferences of the RBP in question (Figure S6). Additional model performance metrics and CNN filter motifs are available (Table S2, Figure S7-S30). Alternatively, filters can also be produced from sequences scoring above a certain score, suc as 0.5 (Figure S31).

DeepCLIP predictions and binding profiles explain splicing mutations

Splicing of mRNA is regulated by binding of RBPs to the nascent pre-mRNA. To test DeepCLIP's ability to predict effects of nucleotide variants on splicing, we generated new models for the splicing factors hnRNP A1 and SRSF1 based on previous CLIP studies (49,51) with DeepCLIP-generated background sequences in order to demonstrate DeepCLIP's performance on novel datasets. We ran 10-fold cross-validation (Figure S32 and S33) using the same input parameters as previously, and extracted the two best performing models. Their pseudo-PFMs are shown in Figure 2a and Figure 2b and they show high visual agreement with previously published motifs of hnRNP A1 and SRSF1 (60,69,71,72). We used the models to predict binding of hnRNP A1 and SRSF1, respectively. We then used the best performing model to predict binding of hnRNP A1 to a set of exonic point mutations (2), grouped into mutations known to cause skipping and mutations known to not cause skipping (Figure 2c-d, Table S6). In Figure 2c, we show the distributions of predictions given the 15-mer sequences used by Raponi et al. in their work describing splicing (2). This specific sequence length represents the length of a typical RNA oligonucleotide used in affinity purification experiments to measure the binding of protein to RNA. Interestingly, we observe no significant difference in the overall change of hnRNP A1 scores for skipping and non-skipping mutations (Figure 2c, left, Wilcoxon signed rank $p = 0.29$), suggesting that hnRNP A1 is not a general regulator of these splicing events. This is to be expected, since only a subset of these events is likely to be meditated by altered binding of hnRNP A1.

When scoring the same 15 nt oligonucleotides with the best performing SRSF1 model, we observe that the SRSF1 scores are significantly different between the two groups (Figure 2c, middle, Wilcoxon signed rank $p = 0.026$). Also, when combining the hnRNP A1 scores with the SRSF1 scores, we find that the groups were significantly different (Figure 2c, right, Wilcoxon signed rank $p = 0.033$).

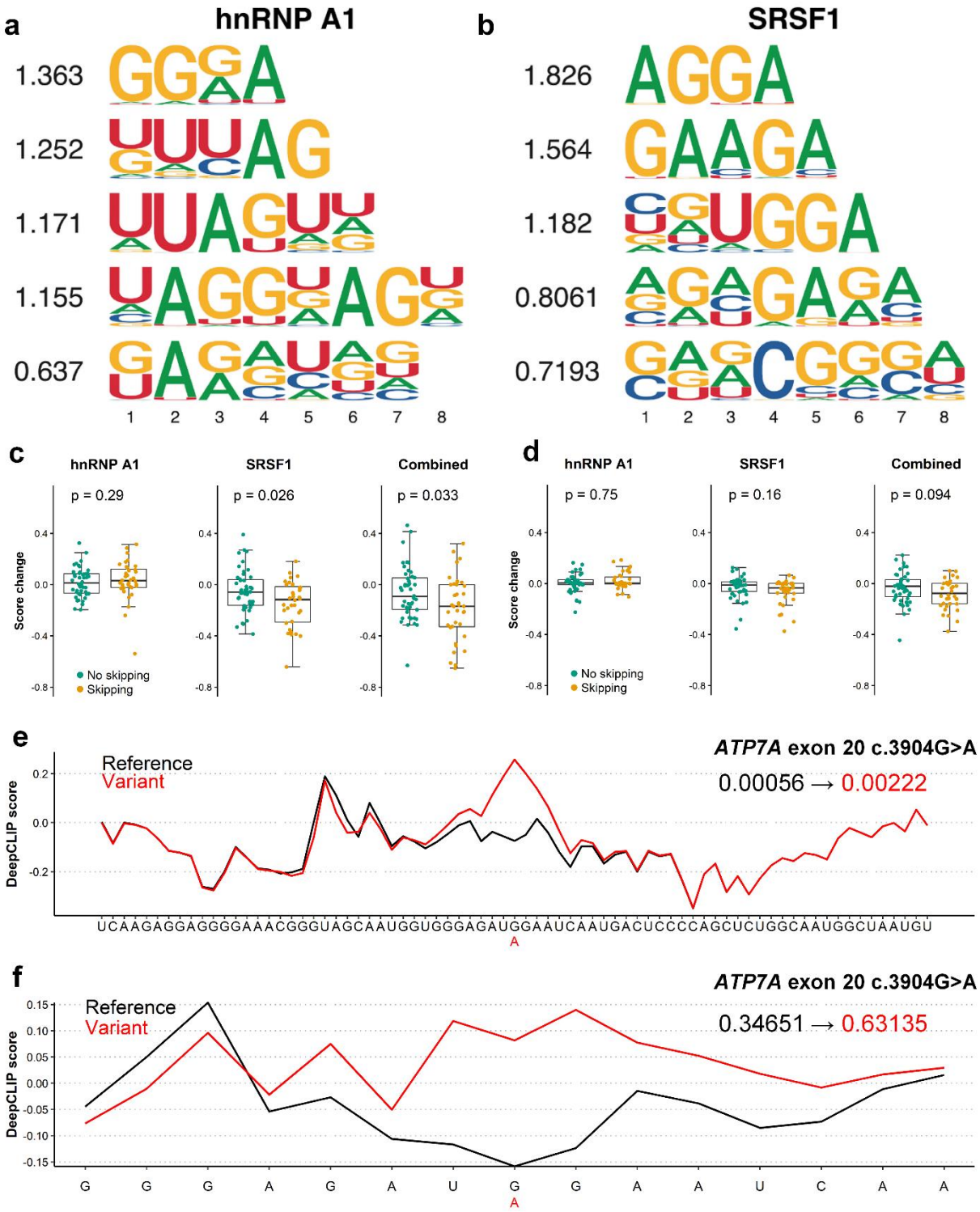


Figure 2 | DeepCLIP models of hnRNPA1 and SRSF1 used to analyze splicing mutations. (a) The CNN filters trained in the hnRNP A1 DeepCLIP model. (b) The CNN filters trained in the SRSF1 model. (c) Box-plots of the distributions of predictions on 15 nt sequences representing wt and mutant versions exons that are skipped upon mutation (yellow, n = 37 sequence-pairs) and exons that remain included in the mutant version (green, n = 46 sequence-pairs) using of the hnRNP A1 model (left), SRSF1 model (middle) and combined (right). The combined change in DeepCLIP scores is obtained by subtracting the hnRNP A1 scores from the SRSF1 scores. Two-tailed Wilcoxon rank sum test p-value is indicated above. Box-plot elements are defined as center line: median, box limits: upper and lower quartiles, whiskers: 1.5x interquartile range. All data points are shown, outliers are not highlighted. (d) Same as (c) but with 75 nt sequences. (e) DeepCLIP binding profile of wt (black) and mutant (red) of the 75 nt sequence representing the *ATP7A* exon 20 +103G>A mutation. The overall DeepCLIP prediction scores are indicated in bold within the plot. (f) Same as (e), but with 15 nt input sequence.

Importantly, the scores of the mutations known to cause skipping were decreased, consistent with the known role of SRSF1 as a positive regulator of exon inclusion. When we used all trained models to analyze the dataset and adjusting p-values for multiple testing, none of the models showed a significant difference in score change between skipping and non-skipping exonic mutations. However, the two SRSF1 models had the lowest adjusted p-values (Figure S34a).

To investigate whether the hnRNP A1 and SRSF1 models improve with extended sequence context, we expanded the 15-mer sequences from the middle and out to a length of 75 nt, the maximum sequence length used during model training. This resulted in less pronounced changes that were not statistically significant (Figure 2d, Table S6), although the combined score indicated that the combined effects of losing SRSF1 binding and gaining hnRNP A1 binding was retained to a higher degree. This is likely caused by DeepCLIP's classification being based on the total binding profile resulting in diminished differences as the sequence is expanded. This is exemplified by the *ATP7A* c.3904G>A exon skipping mutation located at +103 in exon 20 of *ATP7A*, which results in an overall score change of +0.00166 between the wt (0.00056) and mutant (0.00222) 75 nt long sequences (Figure 2e), but a score change of +0.28484 (from 0.34651 to 0.63135) when the 15 nt long sequence is used (Figure 2f). Importantly, both binding profiles predict a localized increase in hnRNP A1 binding to the mutant, showcasing the relevance of using binding profiles when analyzing sequence data and not simply an overall prediction score.

DeepCLIP hnRNP A1 and SRSF1 prediction scores correlate with exon inclusion levels of a known SRSF1-dependent exon

We have previously characterized splicing of *ACADM* exon 5, which shares sequence similarity with *SMN1* exon 7 and we identified a similar regulation with splicing ultimately relying on the balance of SRSF1 and hnRNP A1 binding (34). A prevalent disease-causing c.362C>T mutation reduces the strength of a SRSF1 binding ESE, allows hnRNPA1 binding and causes exon 5 skipping. To test DeepCLIP models of hnRNP A1 and SRSF1 in relation to splicing of *ACADM* exon 5, we generated minigenes with all possible variants at positions c.361, c.362, and c.363 located down-stream of the CAG core motif (Figure 3a). DeepCLIP scores were obtained from the sequences using a window of 36 nt on each side of the 3 positions, totaling 75 nt (Table S7). We then transfected the minigenes in HEK293 cells and measured exon inclusion levels (PSI) using RT-PCR and gel-electrophoresis (Figure 3b). We observe a strong negative correlation (Spearman's ρ = -0.939, p < 2e-16, Figure 3c) between the hnRNP A1 prediction score and the observed inclusion of *ACADM* exon 5, and a strong correlation between the SRSF1 prediction score and the observed inclusions (Spearman's ρ = 0.770, p = 0.0137, Figure 3d). We did not observe a significant correlation with the SRSF1 model trained on the GraphProt benchmark dataset (Figure S35a), which illustrates that the training data impacts the quality of the models. The observed correlation is also very strong for the combined scores (Spearman's ρ = 0.915, p = 0.000467, Figure 3e), in agreement with the hypothesis that the overall inclusion level is a result of the balance between positive and negative factors. The same is true when we use the SRSF1 model generated from the GraphProt dataset (Figure S35b), but not when we perform the same analysis with EX-SKIP (2) on the full exon (Spearman's ρ = -0.177, p = 0.625, Fig S36a). When we use SPANR (1) we do observe a stronger positive correlation than with EX-SKIP

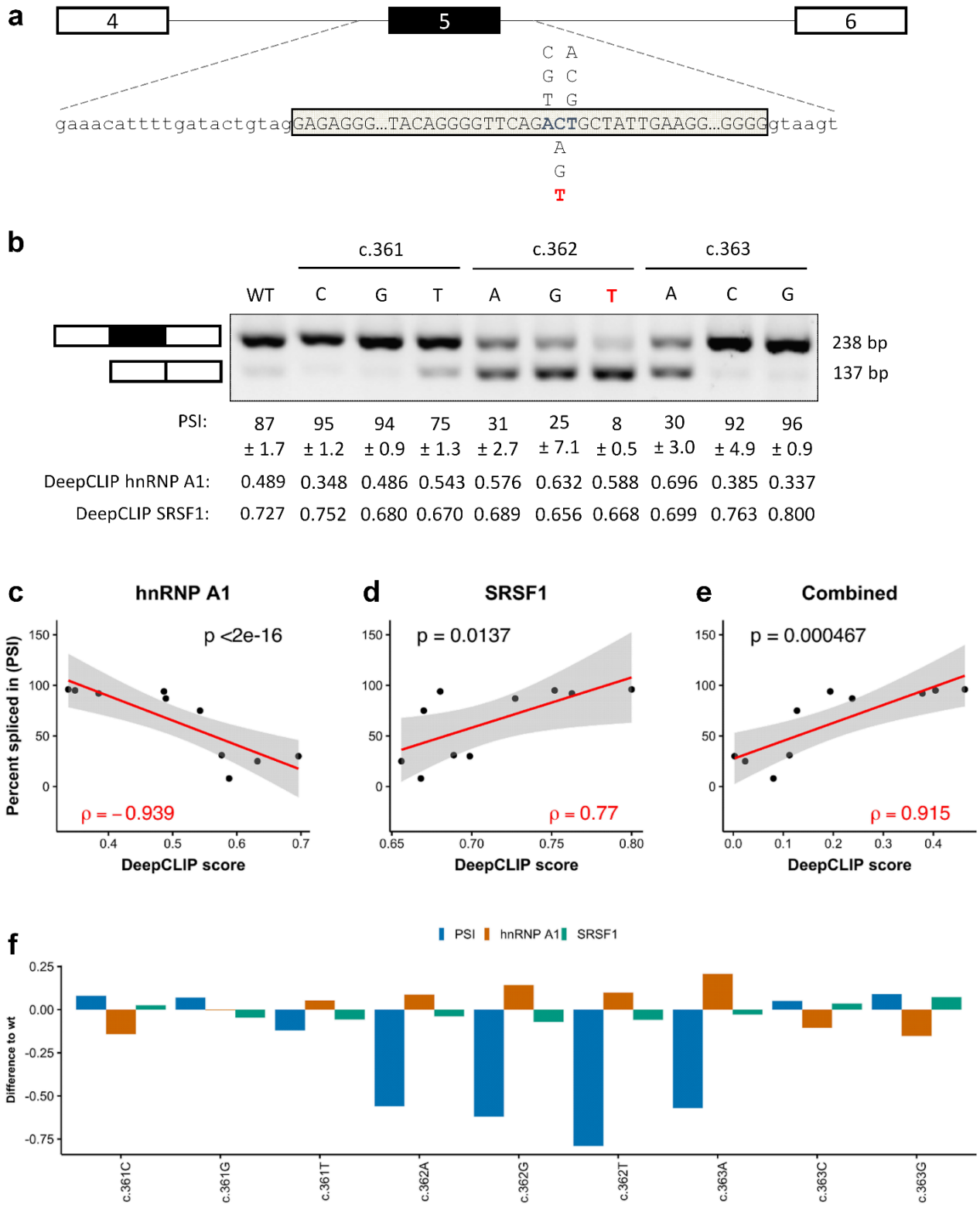


Figure 3 | DeepCLIP models successfully model *ACADM* minigene splicing results. (a) Minigene schematic and location of variants tested, reference in blue. The disease-causing mutation is indicated in red. (b) Splicing of minigenes determined by RT-PCR. Estimates of mean PSI (n = 3) is indicated below, along with 95% CI size. (c) Scatter plot of PSI and DeepCLIP hnRNP A1 score with linear regression (red line, n = 10) and 95% confidence interval (shaded area). (d) Same as (c) but with DeepCLIP SRSF1 score instead. (e) Same as (c) and (d), but showing the DeepCLIP SRSF1 score minus the DeepCLIP hnRNP A1 score. (f) Barplot showing the difference to wt for the minigene PSI and DeepCLIP prediction scores for hnRNP A1 and SRSF1. Spearman's correlation coefficient is indicated in (c), (d), and (e).

(Spearman's $\rho = 0.552$, $p = 0.104$, Fig S36b), but all variants are predicted to have an inclusion level between 81.9% and 82.8%. This conflicts directly with the observed level of exon skipping induced by the disease-causing c.362C>T mutation in patient cells (34) as well as with the observed splicing pattern of the minigenes tested.

Using GraphProt and iDeepS models trained on the SRSF1 eCLIP and hnRNP A1 iCLIP datasets, we could obtain similar but less pronounced correlations, with GraphProt (Figure S36c) performing better than iDeepS (Figure S36d). Overall, when the hnRNP A1 DeepCLIP model predicts an increase in hnRNP A1 binding, there was a decrease in exon 5 inclusion (Figure 3f). In particular, the high degree of exon skipping of all c.362 variants relative to wt were reflected by increases in hnRNP A1 scores (Figure 3b and f). The c.363A variant is predicted to abolish binding of SRSF1 and increase binding of hnRNP A1 and the minigene analysis demonstrates predominant skipping in agreement with this. Like hnRNP A1, many of the variants predicted to lose SRSF1 binding show increased skipping consistent with a loss of ESE activity. The data indicate that while single DeepCLIP models capture binding preferences of individual proteins, the scores are additive and can be used to model effects of multiple proteins interacting in antagonistic and synergistic ways. Because splicing is complex, multiple other factors may also contribute to the outcome of splicing, which may explain why there is not a complete correlation between scores and observed inclusion levels.

DeepCLIP binding profiles can guide the design of therapeutic antisense oligonucleotides

DeepCLIP models produce binding profiles, which are directly used for prediction calculations. We wanted to test how the binding profiles reflect *in vivo* sequence-protein binding dynamics and see if the profiles can help locating sites where splice-switching oligonucleotides (SSOs) can be applied for correction of splicing. Thus, we analyzed a known disease-causing mutation (73) in the *ACADM* gene, c.468+7A>G. The mutation is located outside the core U1 snRNP binding motif but is located within a larger GT-rich region, which is extended by the A>G mutation suggesting that it could be generating or strengthening a TDP-43 binding site. We selected wt and mutant sequences 75 nt of length by including the first 37 nt from either side of the position of the A>G mutation. We then generated a TDP-43 DeepCLIP model based on publicly available binding sites from the POSTAR2 database (50) using the same model parameters as previously (Figure S37), and scored the wt and mutant sequences. We found that DeepCLIP predicts increased binding of TDP-43 to the mutant relative to the wt (Figure 4a).

Next, we designed a minigene harboring *ACADM* exon 6 and part of the flanking introns to test whether the c.468+7A>G mutation affects splicing of exon 6. We found that the mutation caused dramatic skipping of exon 6 from the minigene (Figure 4b). We hypothesized that this was caused by an increase of TDP-43 binding to the mutant sequence, and that exon skipping therefore could be reversed by treating the cells containing the minigenes with siRNA targeting TDP-43 mRNA. Indeed, TDP-43 siRNA treatment resulted in increased exon inclusion (Figure 4b), corroborating that the c.468+7A>G mutation generates a TDP-43 binding site that causes skipping of *ACADM* exon 6. Splice-switching oligonucleotides (SSOs) are a type of antisense oligonucleotide (ASO) that can be used to modulate splicing by sterically preventing binding of splicing regulatory factors to the RNA.

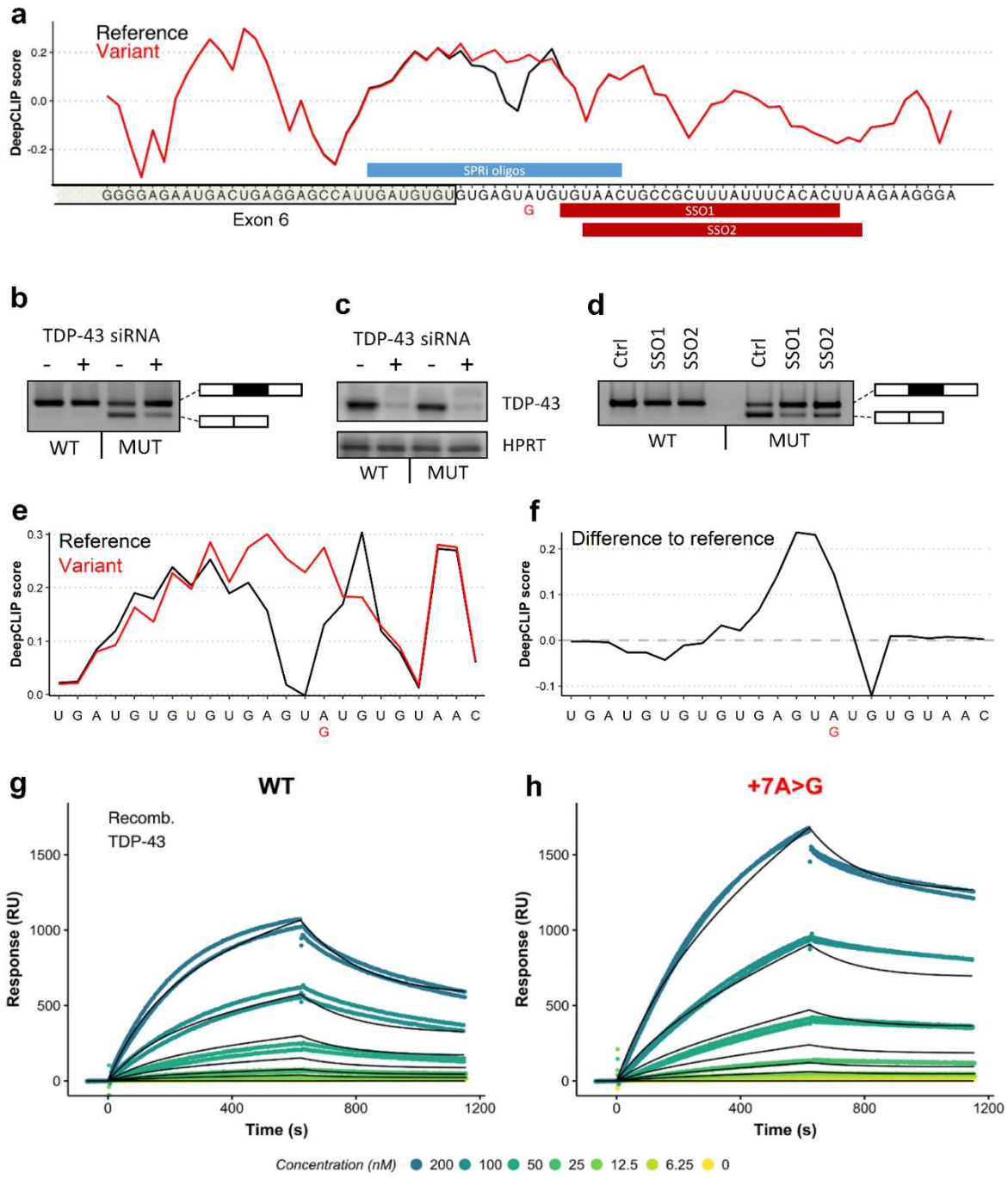


Figure 4 | DeepCLIP predicts increased TDP43 binding as mechanism behind *ACADM* exon 6 skipping. (a) DeepCLIP TDP43 profile across the 5'ss of *ACADM* exon 6 with wt indicated in black and patient mutation indicated in red. Along the first axis the sequence is shown and along the second axis the DeepCLIP BLSTM values are shown. SPRi oligo location and SSO locations are indicated in blue and red bars above and below the sequence, respectively. (b) Splicing of wt and mutant minigenes with either TDP-43 targeting siRNA or non-targeting siRNA determined by RT-PCR. (c) Western blot of TDP-43 and HPRT from siRNA and minigene transfected samples. (d) Splicing of wt and mutant minigenes treated with either a control SSO (Ctrl-SSO), SSO1, or SSO2 determined by RT-PCR. (e) DeepCLIP profile of short RNA oligos used in SPRi measurement, reference in black and +7A>G variant in red. (f) The difference in DeepCLIP binding profiles in (e) between reference and variant. Positive score indicates higher score in variant. (g) SPRi measurements of TDP-43 binding to the wt oligo in (e). (h) SPRi measurements of TDP-43 binding to the variant oligo in (e). In both (g) and (h) the black line indicates the fitted binding model.

Because of the close proximity of the mutation to the 5' splice site, directly blocking the mutant position with an SSO most likely would not result in increased exon inclusion. Interestingly, DeepCLIP finds sites important for binding in a region downstream of the GT-rich core binding motif, which suggests that blocking these sites could prevent TDP-43 binding to the core motif and restore splicing of exon 6. We tested this hypothesis using two different SSO molecules that targeted this downstream region and which had small overlaps with the end of the GT-rich region (Figure 4d). Strikingly, both SSOs proved very efficacious and almost completely restored splicing from the mutant minigene, indicating that blocking of the downstream motif prevented binding of TDP-43. This indicates that TDP-43 may exhibit context dependent binding modularity, and that the DeepCLIP model is able to detect these context-dependent signatures from the sequence alone.

To validate that TDP-43 binding is directly affected by the mutation, we first analyzed a set of 23 nt oligonucleotides with DeepCLIP (Figure 4e,f) showing that in this shorter context the mutation is still predicted to increase. We then used Surface Plasmon Resonance imaging (SPRi) to measure binding to the 3' biotin labeled RNA oligonucleotides and observed a pronounced increase in TDP-43 binding to the mutant (Figure 4g-h) in agreement with DeepCLIP predictions.

DeepCLIP binding scores correlate with *in vitro* binding affinities

One of the most important tasks of a model that predicts presence of RBP binding sites is to accurately estimate the effects of mutations on binding affinity. Therefore, we analyzed 6 sets of wt and mutant exonic variants from the Raponi et al 15-mer set (2) employing SPRi using recombinant hnRNP A1 and SRSF1 as input-proteins (Figure S38-S43, Table S8). These measurements allow quantification of the binding to wt and mutant oligonucleotides, allowing confirmation of DeepCLIP predictions, such as the increase in hnRNP A1 binding to the *ATP7A* exon 20 c.3904G>A mutant (Figure 5a-b). The maximum affinity values obtained by fitting binding models to the measured response by the different SPRi-models correlated well with both hnRNP A1 and SRSF1 DeepCLIP models (Figure 5c-d), across the diverse set of sequences in the dataset (hnRNP A1: Spearman correlation ρ=0.874, p=0.000309; SRSF1: Spearman correlation ρ=0.77, p=0.0137). This was also true when we compared DeepCLIP predictions with the Rmax value (Figure S44). This demonstrates that despite being trained on *in vivo* data, the modelling approach of DeepCLIP is also applicable with short *in vitro* sequences, which can be used to examine and validate specific changes in binding to target sites identified by DeepCLIP.

DeepCLIP analysis of TDP-43-repressed pseudoexons indicates that tissue-specificity is position-dependent

In addition to analyzing sequence variations, DeepCLIP can also be used on a global scale to conduct larger analyses of binding preferences of RBPs. TDP-43 is depleted in the nucleus of motor neurons in patients suffering from amyotrophic lateral sclerosis (ALS) (74,75). TDP-43 has been reported to repress the inclusion of pseudoexons, and these are then erroneously activated following nuclear depletion, potentially leading to development of ALS symptoms (76). A conditional TDP-43 knock-out mouse-model displays increased pseudoexon inclusion, some of which are muscle and neuron-specific (54). Because these pseudoexons are not necessarily conserved in humans, they may not directly relate to ALS, but they may nevertheless improve our understanding of how some

pseudoexons are selectively up-regulated in motor neurons. This can prove important to the understanding of the underlying molecular pathology of ALS. We therefore used DeepCLIP to analyze TDP-43-repressed pseudoexons in mice to examine the tissue specific differences in TDP-43

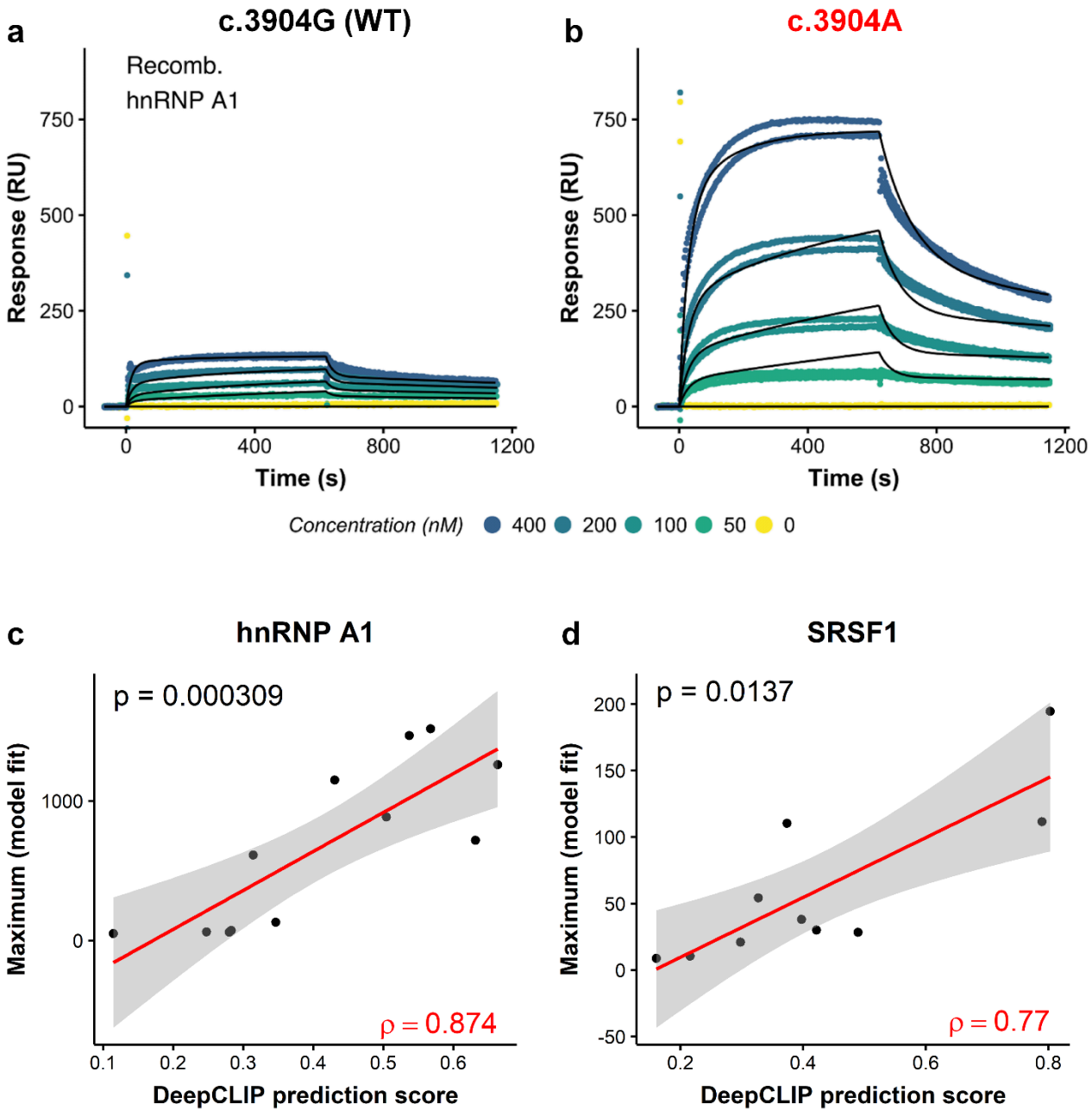


Figure 5 | DeepCLIP predictions correlate with binding affinity studies. (a,b) Scatter plots of raw hnRNP A1 SPRi measurements (dots) and the fitted models (black lines) to wt (a) and mutant (b) *ATP7A* exon 20 15 nt oligonucleotide. (c,d) Scatter plots showing DeepCLIP predictions of hnRNP A1 binding (c) and SRSF1 binding (d) to 15 nt oligonucleotides corresponding to wt and mutant pairs from Raponi et al against the maximum of the binding model fitted to SPRi measurements. The 95% confidence intervals of fitted linear regression models (red line) are shown in grey. Spearman's rho is show in red in lower right corner, and the p-value in the upper left.

binding. We found that DeepCLIP overall predicted decreased binding to the region down-stream of the 5' splice site of pseudoexons that are neuron specific compared to pseudoexons that are muscle-specific (Figure 6, Figure S45), while neuron-specific pseudoexons were predicted to bind more TDP-43 in the region covering the poly-pyrimidine tract compared to muscle-specific pseudoexons. This might reflect interplay between TDP-43 and tissue-specific factors interacting with these regions in a position-dependent manner. These results indicate that sequence analysis of known pseudoexons can lead to discovery of neuron-specific pseudoexons involved in ALS pathology in humans.

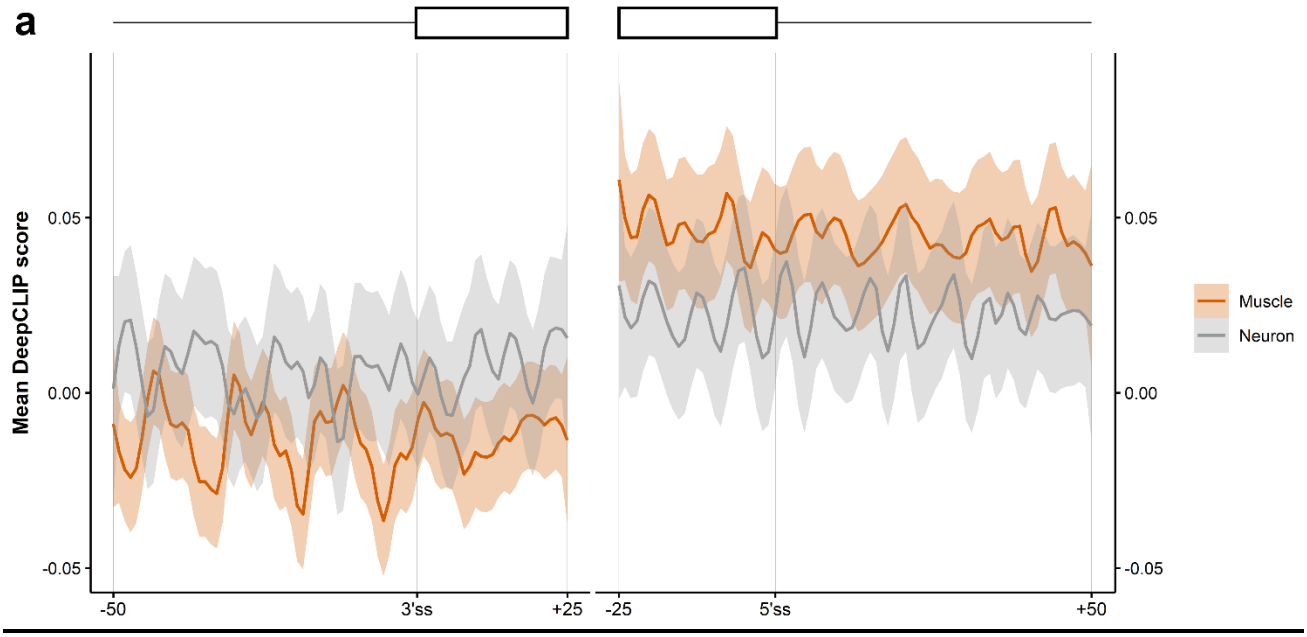


Figure 6 | DeepCLIP analysis of TDP-43-repressed pseudoexons indicate position-dependent tissue-specificity. (a) The average DeepCLIP TDP-43 profile scores of 58 neuron-specific and 79 muscle-specific pseudoexons activated in TDP-43-null mice in the areas covering the 25 first and last nucleotides of the pseudoexon, and the 50 nt spanning intronic regions. 95%-confidence intervals are indicated by shaded areas.

DISCUSSION

We present DeepCLIP, a novel deep learning approach to modelling RNA-binding protein sites using a shallow neural network composed of CNN and LSTM layers to capture context-dependent binding. DeepCLIP generalizes well across a diverse set of sequences in both *in vitro* and *in vivo* settings, and produces a profile of the sequence, which indicates sequence elements important for the binding of the RNA binding protein in question.Previous RNA-protein binding classifiers attempted to improve their performance by incorporating context dependencies in a number of different ways, e.g. secondary and tertiary structure, known binding sites of other RNA-binding proteins, and annotated gene regions such as exons, introns, and UTRs.

With DeepCLIP, we demonstrate that a neural network, in which context dependency is not pre-defined, but modelled implicitly by a BLSTM layer, is competitive or outperforming existing classifiers that, in addition to the RNA-sequences, depend on one or more predefined data sets

containing different categories of contextual information. This allows DeepCLIP to be agnostic with regard to other inputs and makes it robust towards any limitations in e.g. the modelling of the structure, or the level and quality of annotation of gene structure and other protein binding sites. Another benefit is the omission of a lengthy structure modeling step. We compared the runtime of DeepCLIP, iDeepS and GraphProt using the positive sequences from the GraphProt benchmark sets as input to their respective models and found that DeepCLIP was app. 250 faster than GraphProt, and app. 50 times faster than iDeepS based on linear regression of their running times (Figure S46). This greatly improves the scalability of RBP analysis, enabling larger transcriptomic regions to be analyzed by DeepCLIP in long prediction mode.

Secondary RNA structure modelling was previously shown to improve model accuracy for the datasets Ago1-4, CAPRIN1, IGF2BP1-3, MOV10 and ZC3H7B (14), and in general RBPs preferentially bind to structured RNA (77). While DeepCLIP does not directly include predictions of secondary structures when classifying, DeepCLIP AUROC measures for these proteins were the highest of all classifiers except for IGF2BP1-3, where iONMF, which also does not model structure, had a higher AUROC score. This indicates that the BLSTM layer of DeepCLIP captures contextual dependencies that are as important as secondary RNA structure modelling. The inclusion of tertiary structure modelling improved the performance of mDBN+ on the hnRNP C, PTBP1 and TDP-43 datasets over DeepCLIP, indicating that more advanced structure modelling provides an improvement in the prediction of some proteins.

DeepCLIP produces motifs of varying sizes ranked by the average information content (Figure 2a, b). The top-ranking motifs of DeepCLIP for the analyzed proteins were visually remarkably similar to core binding sites as described in literature (67,68,72,78-82) (Figure S6). The motifs depicted in Figure S6 are based on the top-1000 scoring sequences from the positive and negative input dataset and represent the two pseudo-PFMs with the highest mean information score among the five pseudo-PFMs produced. In addition, DeepCLIP allows pseudo-PFM generation based on sequences scoring above a given threshold, e.g. sequences with score higher than 0.5 (see Figures S31), which may produce slightly different pseudo-PFMs. This is a result of the more heterogeneous sequence input when using more of the training input, which results in visualizations with less conserved motifs, but the underlying CNN filters remain the same. DeepCLIP does not model or generate motifs describing structural preferences of RBPs. However, this may be obtained by using structure modeling software on high-scoring sites identified by DeepCLIP.

When searching for binding sites in longer sequences, the information contained in the DeepCLIP binding profiles becomes invaluable, since it unravels interesting areas that are important for protein binding (Figure 2e-f). In the case where a longer sequence contains mainly strong background patterns and only a small segment with binding site potential, DeepCLIP and other tools will be prone to classify this sequence as a background sequence. However, DeepCLIP is able to identify the foreground segment and highlight it on the binding profile of the sequence.

DeepCLIP binding profiles can be used for estimating high- and low-affinity regions of sequences (Figure 4). The prediction scores, which are directly based on the binding profile values, display a strong correlation with affinity studies (Figure 5) suggesting that DeepCLIP successfully captures binding preferences of RBPs. To this end, the binding profiles produced by DeepCLIP can be used to identify splicing regulatory sites that can be targeted by SSOs (Figure 4a), which is an important novel

and missing functionality of existing binding site discovery tools. Thus, DeepCLIP greatly facilitates the design of new drugs based on blocking protein-RNA binding sites, which is a very promising new therapeutic approach, as illustrated for instance by the recent success of the SpinrazaTM SSO in treating SMA (83,84). DeepCLIP could potentially be used to predict binding of RBPs to SSOs themselves (Table S9). However, as SSOs are chemically modified, their binding properties are not directly comparable to RNA and the predictions based on studies of RNA-protein binding are only very uncertain estimates.

DeepCLIP models are trained on protein-RNA binding data from studies on specific proteins and cannot be used to predict binding preferences of proteins where no such data yet exists. Because the binding properties of both proteins and RNA depend on their sequence, it may be possible to train networks that predict binding based on protein and RNA sequences in pairs, such that novel interactions could be predicted from unknown RBPs. Several approaches for this exist, such as autoencoders or generative adversarial networks (GAN). However, some RBPs bind RNA sequence patterns and structures in a more unspecific manner, making it difficult to accurately predict possible binding sites. The utility of this approach needs thorough and rigorous investigation and testing.

DeepCLIP models also depend on the quality of the training data. Significant methodological biases can result in models that recognize more generally CLIP sites than protein specific sites. This is, however, a common trait for the methods and not just DeepCLIP as evidenced by overall similar performance on independent datasets (Figure S5a and b). An example of this is the PUM2 model, which also produces many high prediction scores when used on the positive sequences from QKI, while the QKI model is much less biased (Figure S47a and b). However, because DeepCLIP produces binding profiles we can still use these models to identify potential targets. Using DeepCLIP's long prediction mode we analyzed a transcript, which is known to bind both QKI and PUM2, and defined binding sites as 9 nt windows or more with a mean profile score above 0.3. We then mapped these sites onto the transcript along with known PAR-CLIP and eCLIP sites and found that the PUM2 model identified 4 binding site that were all overlapped by known CLIP sites, either PAR-CLIP or eCLIP. None of them overlapped the QKI sites, and were all located within the 3'UTR, consistent with PUM2's known binding preference for 3'UTR regions. This demonstrates that despite bias in the model, the profile produced still allows for specific identification of PUM2 sites, in the presence of known QKI sites.

In summary, DeepCLIP models provide valuable insight into the functional consequences of sequence variants. Both *in vitro* binding assays and *in vivo* splicing assays as well as observed splicing of disease-causing mutations in patients cells correlate well with DeepCLIP predictions. This demonstrates that an *in silico* analysis with DeepCLIP can serve as a valuable tool for assessing the functional effects of potentially pathogenic sequence variants, providing an important tool for clinical diagnosis. Finally, we demonstrate that DeepCLIP can serve as tool for designing efficient SSOs for correcting aberrant splicing caused by disease-causing mutations. DeepCLIP is freely available, both as stand-alone and as a webtool. The webtool allows analysis of sequences and sequence variants with multiple models in one analysis, as many RBPs compete for binding site positions on RNA molecules, and identifying the most likely change or binding partner is important for further experimental analysis.

FUNDING

This work was supported by grants to TKD from Lundbeckfonden [R231-2016-2823]; and Muskelsvindfonden, and grants to BSA from Natur og Univers, Det Frie Forskningsråd [4181-00515]; Novo Nordisk Fonden (DK) [NNF17OC0029240]; and from ODEx at SDU. The work of AGBG and JB has been supported by VILLUM Young Investigator [grant nr. 73528]. Parts of JB's work was also funded by H2020 project RepoTrial [nr. 777111]. Furthermore, node hours on the Abacaus 2.0 HPC was provided by the DeiC National HPC Centre, SDU.

ACKNOWLEDGEMENTS

We are grateful to laboratory technicians Kasper Stonor Werner, Susanne Ruszczycka, and Gabriella Jensen for their assistance with experiments.

AUTHOR CONTRIBUTIONS

TKD conceived and designed the project. The core algorithm (intrinsic neural network functionalities, layer inter-connectivities and general network composition) was designed and implemented by AGBG. DeepCLIP was implemented by AGBG and TKD. The DeepCLIP web-interface was designed and implemented by SJL. Experiments were designed by TKD, AGBG and BSA and performed by AGBG, USSP, LLH, GHB, MBH, and AMH. All computational analyses were designed by AGBG, TKD, BSA, and JB, and performed by AGBG and TKD. All authors read and contributed to the manuscript.

COMPETING INTERESTS

The authors declare no competing interests, financial or otherwise.

DATA AVAILABILITY

All data, including raw data for all figures, in this study is available either through http://github.com/deepclip or http://deepclip.compbio.sdu.dk or direct communication with the authors.

CODE AVAILABILITY

All code is available at http://github.com/deepclip

REFERENCES

1. Xiong, H.Y., Alipanahi, B., Lee, L.J., Bretschneider, H., Merico, D., Yuen, R.K., Hua, Y., Gueroussov, S., Najafabadi, H.S., Hughes, T.R. *et al.* (2015) RNA splicing. The human splicing code reveals new insights into the genetic determinants of disease. *Science*, **347**, 1254806.
2. Raponi, M., Kralovicova, J., Copson, E., Divina, P., Eccles, D., Johnson, P., Baralle, D. and Vorechovsky, I. (2011) Prediction of single-nucleotide substitutions that result in exon skipping: identification of a splicing silencer in BRCA1 exon 6. *Hum Mutat*, **32**, 436-444.
3. Shibata, A., Okuno, T., Rahman, M.A., Azuma, Y., Takeda, J., Masuda, A., Selcen, D., Engel, A.G. and Ohno, K. (2016) IntSplice: prediction of the splicing consequences of intronic single-nucleotide variations in the human genome. *J Hum Genet*, **61**, 633-640.
4. Bruun, G.H., Bang, J.M.V., Christensen, L.L., Broner, S., Petersen, U.S.S., Guerra, B., Gronning, A.G.B., Doktor, T.K. and Andresen, B.S. (2018) Blocking of an intronic splicing silencer completely rescues IKBKAP exon 20 splicing in familial dysautonomia patient cells. *Nucleic Acids Res*, **46**, 7938-7952.
5. Sinha, R., Kim, Y.J., Nomakuchi, T., Sahashi, K., Hua, Y., Rigo, F., Bennett, C.F. and Krainer, A.R. (2018) Antisense oligonucleotides correct the familial dysautonomia splicing defect in IKBKAP transgenic mice. *Nucleic Acids Res*, **46**, 4833-4844.
6. Hua, Y., Vickers, T.A., Okunola, H.L., Bennett, C.F. and Krainer, A.R. (2008) Antisense masking of an hnRNP A1/A2 intronic splicing silencer corrects SMN2 splicing in transgenic mice. *Am J Hum Genet*, **82**, 834-848.
7. Ule, J., Jensen, K., Mele, A. and Darnell, R.B. (2005) CLIP: a method for identifying protein-RNA interaction sites in living cells. *Methods*, **37**, 376-386.
8. Hafner, M., Landthaler, M., Burger, L., Khorshid, M., Hausser, J., Berninger, P., Rothballer, A., Ascano, M., Jr., Jungkamp, A.C., Munschauer, M. *et al.* (2010) Transcriptome-wide identification of RNA-binding protein and microRNA target sites by PAR-CLIP. *Cell*, **141**, 129-141.
9. König, J., Zarnack, K., Rot, G., Curk, T., Kayikci, M., Zupan, B., Turner, D.J., Luscombe, N.M. and Ule, J. (2010) iCLIP reveals the function of hnRNP particles in splicing at individual nucleotide resolution. *Nature structural & molecular biology*, **17**, 909-915.
10. Bailey, T.L., Williams, N., Misleh, C. and Li, W.W. (2006) MEME: discovering and analyzing DNA and protein sequence motifs. *Nucleic Acids Res*, **34**, W369-373.
11. Heinz, S., Benner, C., Spann, N., Bertolino, E., Lin, Y.C., Laslo, P., Cheng, J.X., Murre, C., Singh, H. and Glass, C.K. (2010) Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities. *Mol Cell*, **38**, 576-589.
12. Kazan, H., Ray, D., Chan, E.T., Hughes, T.R. and Morris, Q. (2010) RNAcontext: a new method for learning the sequence and structure binding preferences of RNA-binding proteins. *PLoS Comput Biol*, **6**, e1000832.
13. Hiller, M., Pudimat, R., Busch, A. and Backofen, R. (2006) Using RNA secondary structures to guide sequence motif finding towards single-stranded regions. *Nucleic Acids Res*, **34**, e117.
14. Maticzka, D., Lange, S.J., Costa, F. and Backofen, R. (2014) GraphProt: modeling binding preferences of RNA-binding proteins. *Genome Biol*, **15**, R17.
15. Steffen, P., Voss, B., Rehmsmeier, M., Reeder, J. and Giegerich, R. (2006) RNAshapes: an integrated RNA analysis package based on abstract shapes. *Bioinformatics*, **22**, 500-503.
16. Zhang, X.H. and Chasin, L.A. (2004) Computational definition of sequence motifs governing constitutive exon splicing. *Genes & development*, **18**, 1241-1250.

17. Fairbrother, W.G., Yeh, R.F., Sharp, P.A. and Burge, C.B. (2002) Predictive identification of exonic splicing enhancers in human genes. *Science*, **297**, 1007-1013.
18. Strazar, M., Zitnik, M., Zupan, B., Ule, J. and Curk, T. (2016) Orthogonal matrix factorization enables integrative analysis of multiple RNA binding proteins. *Bioinformatics*, **32**, 1527-1535.
19. Henaff, M., Weston, J., Szlam, A., Bordes, A. and LeCun, Y. (2016) Tracking the World State with Recurrent Entity Networks. *CoRR*, **abs/1612.03969**.
20. Redmon, J. and Farhadi, A. (2016) YOLO9000: Better, Faster, Stronger. *CoRR*, **abs/1612.08242**.
21. Helmstaedter, M., Briggman, K.L., Turaga, S.C., Jain, V., Seung, H.S. and Denk, W. (2013) Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, **500**, 168-174.
22. LeCun, Y., Bengio, Y. and Hinton, G. (2015) Deep learning. *Nature*, **521**, 436-444.
23. Leung, M.K., Xiong, H.Y., Lee, L.J. and Frey, B.J. (2014) Deep learning of the tissue-regulated splicing code. *Bioinformatics*, **30**, i121-i129.
24. Quang, D. and Xie, X. (2016) DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences. *Nucleic Acids Res*, **44**, e107.
25. Hill, S.T., Kuintzle, R., Teegarden, A., Merrill, E., 3rd, Danaee, P. and Hendrix, D.A. (2018) A deep recurrent neural network discovers complex biological rules to decipher RNA protein-coding potential. *Nucleic Acids Res*.
26. Almagro Armenteros, J.J., Sønderby, C.K., Sønderby, S.K., Nielsen, H. and Winther, O. (2017) DeepLoc: prediction of protein subcellular localization using deep learning. *Bioinformatics*, **33**, 3387-3395.
27. Alipanahi, B., Delong, A., Weirauch, M.T. and Frey, B.J. (2015) Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat Biotechnol*, **33**, 831-838.
28. Ben-Bassat, I., Chor, B. and Orenstein, Y. (2018) A deep neural network approach for learning intrinsic protein-RNA binding preferences. *Bioinformatics*, **34**, i638-i646.
29. Zhang, S., Zhou, J., Hu, H., Gong, H., Chen, L., Cheng, C. and Zeng, J. (2016) A deep learning framework for modeling structural features of RNA-binding protein targets. *Nucleic Acids Res*, **44**, e32.
30. Pan, X. and Shen, H.B. (2017) RNA-protein binding motifs mining with a new hybrid deep learning based cross-domain knowledge integration approach. *BMC bioinformatics*, **18**, 136.
31. Pan, X., Rijnbeek, P., Yan, J. and Shen, H.B. (2018) Prediction of RNA-protein sequence and structure binding preferences using deep convolutional and recurrent neural networks. *BMC Genomics*, **19**, 511.
32. Cartegni, L., Chew, S.L. and Krainer, A.R. (2002) Listening to silence and understanding nonsense: exonic mutations that affect splicing. *Nat Rev Genet*, **3**, 285-298.
33. Huppertz, I., Attig, J., D'Ambrogio, A., Easton, L.E., Sibley, C.R., Sugimoto, Y., Tajnik, M., Konig, J. and Ule, J. (2014) iCLIP: protein-RNA interactions at nucleotide resolution. *Methods*, **65**, 274-287.
34. Nielsen, K.B., Sorensen, S., Cartegni, L., Corydon, T.J., Doktor, T.K., Schroeder, L.D., Reinert, L.S., Elpeleg, O., Krainer, A.R., Gregersen, N. *et al.* (2007) Seemingly neutral polymorphic variants may confer immunity to splicing-inactivating mutations: a synonymous SNP in exon 5 of MCAD protects from deleterious mutations in a flanking exonic splicing enhancer. *Am J Hum Genet*, **80**, 416-432.
35. Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2012) ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, **25**.
36. LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W. and Jackel, L. (1989), *Neural Information Processing Systems (NIPS)*.
37. Hochreiter, S. and Schmidhuber, J. (1997) Long short-term memory. *Neural Comput*, **9**, 1735-1780.

38. Schuster, M. and Paliwal, K.K. (1997) Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, **45**, 2673-2681.
39. Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., Bastien, F., Bayer, J. and Team, T.D. (2016) Theano: A {Python} framework for fast computation of mathematical expressions. *arXiv e-prints*, **abs/1605.02688**.
40. Dieleman, S., Schlüter, J., Raffel, C., Olson, E., Sønderby, S.K., Nouri, D., Maturana, D., Thoma, M., Battenberg, E., Kelly, J. *et al.* (2015).
41. Goldberg, Y. (2016) A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research*, **57**, 345–420.
42. Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep Learning*. MIT Press.
43. Nair, V. and Hinton, G.E. (2010), *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807-814.
44. Srivastava, R.K., Masci, J., Kazerounian, S., Gomez, F. and Schmidhuber, J.r. (2013) Compete to Compute. 2310--2318.
45. Park, S., Min, S., Choi, H. and Yoon, S. (2016) deepMiRGene: deep neural network based precursor microRNA prediction. *arXiv preprint arXiv:1605.00017*.
46. Bahdanau, D., Cho, K. and Bengio, Y. (2014) Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, **abs/1409.0473**.
47. Kingma, D. and Ba, J. (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
48. Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J.C. and Muller, M. (2011) pROC: an open-source package for R and S+ to analyze and compare ROC curves. *BMC bioinformatics*, **12**, 77.
49. Van Nostrand, E.L., Pratt, G.A., Shishkin, A.A., Gelboin-Burkhart, C., Fang, M.Y., Sundararaman, B., Blue, S.M., Nguyen, T.B., Surka, C., Elkins, K. *et al.* (2016) Robust transcriptome-wide discovery of RNA-binding protein binding sites with enhanced CLIP (eCLIP). *Nature methods*, **13**, 508-514.
50. Zhu, Y., Xu, G., Yang, Y.T., Xu, Z., Chen, X., Shi, B., Xie, D., Lu, Z.J. and Wang, P. (2019) POSTAR2: deciphering the post-transcriptional regulatory logics. *Nucleic Acids Res*, **47**, D203-D211.
51. Bruun, G.H., Doktor, T.K., Borch-Jensen, J., Masuda, A., Krainer, A.R., Ohno, K. and Andresen, B.S. (2016) Global identification of hnRNP A1 binding sites for SSO-based splicing modulation. *BMC Biol*, **14**, 54.
52. Ray, D., Kazan, H., Cook, K.B., Weirauch, M.T., Najafabadi, H.S., Li, X., Gueroussov, S., Albu, M., Zheng, H., Yang, A. *et al.* (2013) A compendium of RNA-binding motifs for decoding gene regulation. *Nature*, **499**, 172-177.
53. Hahne, F. and Ivanek, R. (2016) Visualizing Genomic Data Using Gviz and Bioconductor. *Methods Mol Biol*, **1418**, 335-351.
54. Jeong, Y.H., Ling, J.P., Lin, S.Z., Donde, A.N., Braunstein, K.E., Majounie, E., Traynor, B.J., LaClair, K.D., Lloyd, T.E. and Wong, P.C. (2017) Tdp-43 cryptic exons are highly variable between cell types. *Mol Neurodegener*, **12**, 13.
55. Baltz, A.G., Munschauer, M., Schwanhausser, B., Vasile, A., Murakawa, Y., Schueler, M., Youngs, N., Penfold-Brown, D., Drew, K., Milek, M. *et al.* (2012) The mRNA-bound proteome and its global occupancy profile on protein-coding transcripts. *Mol Cell*, **46**, 674-690.
56. Hoell, J.I., Larsson, E., Runge, S., Nusbaum, J.D., Duggimpudi, S., Farazi, T.A., Hafner, M., Borkhardt, A., Sander, C. and Tuschl, T. (2011) RNA targets of wild-type and mutant FET family proteins. *Nature structural & molecular biology*, **18**, 1428-1431.

57. Kishore, S., Jaskiewicz, L., Burger, L., Hausser, J., Khorshid, M. and Zavolan, M. (2011) A quantitative analysis of CLIP methods for identifying binding sites of RNA-binding proteins. *Nature methods*, **8**, 559-564.
58. Lebedeva, S., Jens, M., Theil, K., Schwanhausser, B., Selbach, M., Landthaler, M. and Rajewsky, N. (2011) Transcriptome-wide analysis of regulatory interactions of the RNA-binding protein HuR. *Mol Cell*, **43**, 340-352.
59. Mukherjee, N., Corcoran, D.L., Nusbaum, J.D., Reid, D.W., Georgiev, S., Hafner, M., Ascano, M., Jr., Tuschl, T., Ohler, U. and Keene, J.D. (2011) Integrative regulatory mapping indicates that the RNA-binding protein HuR couples pre-mRNA processing and mRNA stability. *Mol Cell*, **43**, 327-339.
60. Sanford, J.R., Wang, X., Mort, M., Vanduyn, N., Cooper, D.N., Mooney, S.D., Edenberg, H.J. and Liu, Y. (2009) Splicing factor SFRS1 recognizes a functionally diverse landscape of RNA transcripts. *Genome research*, **19**, 381-394.
61. Sievers, C., Schlumpf, T., Sawarkar, R., Comoglio, F. and Paro, R. (2012) Mixture models and wavelet transforms reveal high confidence RNA-protein interaction sites in MOV10 PAR-CLIP data. *Nucleic Acids Res*, **40**, e160.
62. Wang, Z., Kayikci, M., Briese, M., Zarnack, K., Luscombe, N.M., Rot, G., Zupan, B., Curk, T. and Ule, J. (2010) iCLIP predicts the dual splicing effects of TIA-RNA interactions. *PLoS biology*, **8**, e1000530.
63. Xue, Y., Zhou, Y., Wu, T., Zhu, T., Ji, X., Kwon, Y.S., Zhang, C., Yeo, G., Black, D.L., Sun, H. *et al.* (2009) Genome-wide analysis of PTB-RNA interactions reveals a strategy used by the general splicing repressor to modulate exon inclusion or skipping. *Mol Cell*, **36**, 996-1006.
64. Orenstein, Y., Wang, Y. and Berger, B. (2016) RCK: accurate and efficient inference of sequence- and structure-based protein-RNA binding models from RNAcompete data. *Bioinformatics*, **32**, i351-i359.
65. Buratti, E., Brindisi, A., Pagani, F. and Baralle, F.E. (2004) Nuclear factor TDP-43 binds to the polymorphic TG repeats in CFTR intron 8 and causes skipping of exon 9: a functional link with disease penetrance. *Am J Hum Genet*, **74**, 1322-1325.
66. Barker, A., Epis, M.R., Porter, C.J., Hopkins, B.R., Wilce, M.C., Wilce, J.A., Giles, K.M. and Leedman, P.J. (2012) Sequence requirements for RNA binding by HuR and AUF1. *Journal of biochemistry*, **151**, 423-437.
67. Gregersen, L.H., Schueler, M., Munschauer, M., Mastrobuoni, G., Chen, W., Kempa, S., Dieterich, C. and Landthaler, M. (2014) MOV10 Is a 5' to 3' RNA helicase contributing to UPF1 mRNA target degradation by translocation along 3' UTRs. *Mol Cell*, **54**, 573-585.
68. Perez, I., Lin, C.H., McAfee, J.G. and Patton, J.G. (1997) Mutation of PTB binding sites causes misregulation of alternative 3' splice site selection in vivo. *RNA*, **3**, 764-778.
69. Wang, X., Juan, L., Lv, J., Wang, K., Sanford, J.R. and Liu, Y. (2011) Predicting sequence and structural specificities of RNA binding regions recognized by splicing factor SRSF1. *BMC Genomics*, **12 Suppl 5**, S8.
70. Dember, L.M., Kim, N.D., Liu, K.Q. and Anderson, P. (1996) Individual RNA recognition motifs of TIA-1 and TIAR have different RNA binding specificities. *J Biol Chem*, **271**, 2783-2788.
71. Cartegni, L., Wang, J., Zhu, Z., Zhang, M.Q. and Krainer, A.R. (2003) ESEfinder: A web resource to identify exonic splicing enhancers. *Nucleic Acids Res*, **31**, 3568-3571.
72. Feng, H., Bao, S., Rahman, M.A., Weyn-Vanhentenryck, S.M., Khan, A., Wong, J., Shah, A., Flynn, E.D., Krainer, A.R. and Zhang, C. (2019) Modeling RNA-Binding Protein Specificity In Vivo by Precisely Registering Protein-RNA Crosslink Sites. *Mol Cell*, **74**, 1189-1204 e1186.

73. Waddell, L., Wiley, V., Carpenter, K., Bennetts, B., Angel, L., Andresen, B.S. and Wilcken, B. (2006) Medium-chain acyl-CoA dehydrogenase deficiency: genotype-biochemical phenotype correlations. *Mol Genet Metab*, **87**, 32-39.
74. Arai, T., Hasegawa, M., Akiyama, H., Ikeda, K., Nonaka, T., Mori, H., Mann, D., Tsuchiya, K., Yoshida, M., Hashizume, Y. *et al.* (2006) TDP-43 is a component of ubiquitin-positive tau-negative inclusions in frontotemporal lobar degeneration and amyotrophic lateral sclerosis. *Biochem Biophys Res Commun*, **351**, 602-611.
75. Neumann, M., Sampathu, D.M., Kwong, L.K., Truax, A.C., Micsenyi, M.C., Chou, T.T., Bruce, J., Schuck, T., Grossman, M., Clark, C.M. *et al.* (2006) Ubiquitinated TDP-43 in frontotemporal lobar degeneration and amyotrophic lateral sclerosis. *Science*, **314**, 130-133.
76. Ling, J.P., Pletnikova, O., Troncoso, J.C. and Wong, P.C. (2015) TDP-43 repression of nonconserved cryptic exons is compromised in ALS-FTD. *Science*, **349**, 650-655.
77. Sanchez de Groot, N., Armaos, A., Grana-Montes, R., Alriquet, M., Calloni, G., Vabulas, R.M. and Tartaglia, G.G. (2019) RNA structure drives interaction with proteins. *Nat Commun*, **10**, 3246.
78. Gao, F.B., Carson, C.C., Levine, T. and Keene, J.D. (1994) Selection of a subset of mRNAs from combinatorial 3' untranslated region libraries using neuronal RNA-binding protein Hel-N1. *Proc Natl Acad Sci U S A*, **91**, 11207-11211.
79. Munteanu, A., Mukherjee, N. and Ohler, U. (2018) SSMART: sequence-structure motif identification for RNA-binding proteins. *Bioinformatics*, **34**, 3990-3998.
80. Colombrita, C., Onesto, E., Megiorni, F., Pizzuti, A., Baralle, F.E., Buratti, E., Silani, V. and Ratti, A. (2012) TDP-43 and FUS RNA-binding proteins bind distinct sets of cytoplasmic messenger RNAs and differently regulate their post-transcriptional fate in motoneuron-like cells. *J Biol Chem*, **287**, 15635-15647.
81. Adinolfi, M., Pietrosanto, M., Parca, L., Ausiello, G., Ferre, F. and Helmer-Citterich, M. (2019) Discovering sequence and structure landscapes in RNA interaction motifs. *Nucleic Acids Res*, **47**, 4958-4969.
82. Meyer, C., Garzia, A., Mazzola, M., Gerstberger, S., Molina, H. and Tuschl, T. (2018) The TIA1 RNA-Binding Protein Family Regulates EIF2AK2-Mediated Stress Response and Cell Cycle Progression. *Mol Cell*, **69**, 622-635 e626.
83. Finkel, R.S., Mercuri, E., Darras, B.T., Connolly, A.M., Kuntz, N.L., Kirschner, J., Chiriboga, C.A., Saito, K., Servais, L., Tizzano, E. *et al.* (2017) Nusinersen versus Sham Control in Infantile-Onset Spinal Muscular Atrophy. *N Engl J Med*, **377**, 1723-1732.
84. Mercuri, E., Darras, B.T., Chiriboga, C.A., Day, J.W., Campbell, C., Connolly, A.M., Iannaccone, S.T., Kirschner, J., Kuntz, N.L., Saito, K. *et al.* (2018) Nusinersen versus Sham Control in Later-Onset Spinal Muscular Atrophy. *N Engl J Med*, **378**, 625-635.

4 Manuscript 3

Unraveling of the mechanism behind PKG-dependent regulation of NOX4, 5 gene expression for improved ischemic stroke therapy

5 Discussion, outlook and conclusion

This thesis presented three manuscripts that each described a novel bioinformatic program or bioinformatic approach. The development of the programs was motivated by the advent of big data and the analytical possibilities and challenges they introduce. The programs utilized different machine learning methods and techniques, which constitute an essential part of the bioinformatic approach to data analysis. The aims of the manuscripts were to show the new methodologies of the tools and to demonstrate how they can be used to analyze biological data and discover new biological and biomedical insight.

In the second chapter, the manuscript entitled "Comparative single-cell trajectory network enrichment identifies mechanistic patterns in hematopoiesis and CD8 T-cell development" was presented. The manuscript introduced a novel clustering tool for scRNA-seq data called Scellnetor: Single-cell Network Profiler for Extraction of Systems Biology Patterns from scRNA-seq Trajectories. We showed that the tool can cluster gene expression data from single-cells and analyze and compare different user-selectable differentiation trajectories. Based on data from a hematopoiesis study and data from a study on dysfunctional CD8 T-cell development in chronic infections we demonstrated that the tool can find connected subnetworks of genes important for differentiating between the distinct development trajectories.

The methodology of the tool includes the calculation of a hyper-similarity matrix, which is a new approach to clustering when two conditions are to be compared. A number of hyperparameters are involved in the hyper-similarity matrix computation. One of the hyperparameters is the size of a moving average function that smooths the gene expression patterns. We applied the moving average function to reduce the noise of the scRNA-seq data and to increase the similarities of the gene expression patterns of the individual trajectories used in the comparisons. By doing so, we automatically also decreased (or increased) the similarities of the gene expression patterns across the compared trajectories. To support this, we found in preliminary tests that p-values resulting from comparisons of trajectory-related gene expression patterns tended to be lower after data smoothing. We set the default moving average size to 20; a size found after visually inspecting several pseudo-timeline plots generated using different moving average sizes. Whether an optimal size exists could be investigated further, as the size of the moving average affects the resulting clusterings.

Scellnetor dimensionality reduces the scRNA-seq data using UMAP [1], as this improves the clustering of the high-dimensional data. UMAP was chosen for the dimensionality reduction, as it works well on scRNA-seq data and because the runtime is low compared to other dimensionality reduction tools [2]. UMAP allows for the reduction of data to any number of (lower) dimensions, but in the Scellnetor pipeline we reduce the genes' expression patterns to two dimensions. An improvement could be to leave this option to users of Scellnetor, so they can decide

for themselves how much the dimensions of the data should be reduced. However, this introduces a new hyperparameter, which makes using the tool more complex.

PPI networks suffer from technical and study biases, which compromise the reliability of the interactions in the networks. We chose to use the network from BioGRID [3] as our default network, as the interactions are based on curation and supported by experimental validations and findings in the literature. We do not overcome the mentioned biases by choosing curated network, but it reduces the uncertainty associated with the individual interactions. Hub nodes exert a challenge when trying to extract biologically interesting subnetworks using PPI networks, as they seem to find their way into many clusters of clustering approaches that rely on networks [4]. It is difficult to tell whether the hub nodes are caused by study biases or whether these proteins have many connections because they are essential for many cell processes. Our method of reducing the influence of hub nodes in our clusters was to use a constrained agglomerative hierarchical clustering algorithm. Our supposition was that if the dendrogram extension was driven by expression similarity and not directly by network connectivity, the resulting clusters would be based more on within-cluster similarities than on connectivity. But follow up investigations are needed.

A possible drawback of the Scellnetor methodology is that it filters away genes that are not in the default network (or in a user-uploaded network), which reduces the amount of information that can be extracted from the scRNA-seq data. A way to use the filtered genes in the Scellnetor pipeline would be to include them in the hyper-similarity matrix calculation and only exclude them when forming the induced subnetworks. The names and perhaps timelines of the "filtered genes" could be available as additional information next to the relevant clusters. However, as we wanted to find mechanisms (clusters based on interactions), our current approach serves as a good solution. An aim of the Scellnetor-project was to show its utility in precision medicine approaches, and we found interesting connected genes when we analyzed the data from the study on dysfunctional CD8 T-cell development in chronic infections. However, experimental data are needed before it can be confirmed whether or not Scellnetor is able to extra core endophenotypes. Currently, the hyper-similarity matrix allows for the comparisons of two trajectories. However, in theory, the approach can be extended such that several differentiation trajectories can be compared. Moreover, with small modifications the Scellnetor method can be applied to temporal bulk RNA-seq as well as many other data type where different conditions are to be compared. Such extensions might be of relevance to a potential Scellnetor 2.0.

In the third chapter of this thesis, the manuscript entitled "DeepCLIP: Predicting the effect of mutations on protein-RNA binding with Deep Learning" was presented. The manuscript introduced the tool DeepCLIP, which is a convolutional LSTM for RBP motif discovery and binding profile generation. DeepCLIP was developed because we recognized a need to identify how nucleotide variants affect RNA-protein binding and thereby affect vital cellular processes, such as gene expression.

We wanted to implement an easy-to-use bioinformatic tool that can readily estimate RNA sequences' affinity for a range of RBPs. We applied the power of deep learning in our algorithmic solution, as deep learning techniques have delivered state of the art results in several fields and work efficiently with large data sets, such as data sets from CLIP experiments. An aim of the DeepCLIP project was to construct a neural network architecture where all layer outputs could be extracted and meaningfully interpreted in the context of the classification tasks. DeepCLIP produces three types of outputs: DeepCLIP prediction scores, binding profiles and pseudo-PFMs that each can be extracted by different layers in the network. We showed that DeepCLIP predictions correlate well with *in vitro* RBP binding analyses. However, DeepCLIP-predictions do not necessarily reveal whether an investigated sequence is likely to interact with an RBP or not. A DeepCLIP prediction is an estimate of how well an entire sequence resembles the CLIP-sequences that were used in the model training. This means that even though DeepCLIP classifies a sequence as a background sequence, it still is likely to contain a strong binding site for the investigated RBP. Only the binding profiles can reveal the affinity-landscape of the analyzed RBPs. The DeepCLIP binding profile is one of the features that separates the tool from other capable RBP binding analysis tools that also utilize deep learning techniques [5–7]. We showed that DeepCLIP binding profiles excellently capture the binding preferences of RBPs and that they agree with *in vitro* and *in vivo* validation experiments. As a novelty, we showed that the binding profiles can be used to develop treatments that can correct aberrant splicing.

The only other tool that produces binding profiles of RBPs is GraphProt [8]. It is not clear how GraphProt's binding profiles compare to DeepCLIP's, as the GraphProt article itself shows no analyses or examples that use binding profiles. To this end, using GraphProt is more cumbersome compared to DeepCLIP and predicting binding affinities of RNA sequences to RBPs using trained models is much slower. DeepCLIP is, given its intuitive website, easy to use and researchers can quickly generate binding profiles of interesting RBPs. Website functionalities allow for easy comparison of different RBP binding profiles given one or more sequences. The improved usability of DeepCLIP may promote that it gets used in more analysis pipelines and thereby contributes to research.

In DeepCLIP, we implemented a custom-made Winner-Takes-All Enhancer layer that enhances important features found in the preceding connected convolutional layers. This layer was created, because we wanted to "reward" the most important feature found by every convolutional layer while preserving the layers' equivariance to translation. However, it may not be an optimal solution to markedly reward the *one* highest scoring feature found by each of the convolutional layers, as RBPs may have multiple high-affinity binding sites in a single sequence. Only rewarding one site could confuse the network's understanding of the analyzed RBP's binding preferences. The network's performance might improve if the "reward system" was relaxed to include multiple high-scoring binding sites instead

of just one. On the other hand, the use more convolutional layers of different sizes might negate the possible bias.

The output layer of DeepCLIP is special. The node is “dead” or “frozen” - its parameters do not change during training. We found that a frozen output layer “forced” the remaining layers to produce more directly interpretable output values. Very deep neural network layers can build complex data representations out of simpler ones. Thus, constructing a not-so-deep network, like DeepCLIP, may have promoted that the complexity level did not rise to something incomprehensible and, thus, allowed a directly understandable interpretation of layer outputs.

The output layer was a sigmoid node, and a general problem of the sigmoid activation function is that it saturates. For example, if two binding profiles based on two different sequences summed to 50 and 100, respectively, the output layer would output ∼1 in both cases and thereby classify both sequences as “CLIP-sequences”. Even though one of the sequence’s summed profile score is twice as high as the other, the error produced in both cases would be close to identical. This means, the network would be updated by the same amount, even though one sequence might have twice as much affinity for a given RBP as the other. This unnoticed affinity difference may be important when modelling the binding preferences of RBPs and should be accounted for in a potential DeepCLIP 2.0. The development of DeepCLIP was motivated by the ambition to elucidate the mechanisms by which the many newly discovered nucleotide variants might affect cellular processes. The DeepCLIP approach is able to investigate the variants that affect RNA-protein binding, however, the nucleotide variants might affect many other cellular mechanisms, such as DNA-protein binding or induce change of certain amino acid residues.

In the fourth chapter of this thesis, the manuscript entitled “Unraveling of the mechanism behind PKG-dependent regulation of NOX4, 5 gene expression for improved ischemic stroke therapy” was presented. The manuscript provided evidence for a apo-sGCa/PKG-dependent downregulation of NOX4 and NOX5 gene expression during hypoxia. Using a newly developed simulation approach, it was shown that PKG-activity promote downregulation of TFs that otherwise upregulate NOX4 and NOX5. The PKG-dependent downregulation of NOX4 and NOX5 that is shown, is based on *in vitro* experiments. It would benefit the manuscript to include additional validating experiments and to show that the connection exists *in vivo* as well. Additionally, it would be interesting to demonstrate experimentally if PKG-activity promote downregulation of RELA and NF-kB1 as predicted, and if RELA and NF-kB1 in fact do control gene expression of NOX 4, 5 post-stroke.

The simulation approach that is presented in the manuscript depends on several databases. And among them, a single database with information on TF-target connections. The reliability of the simulation might be improved if several combined GRNs were used instead of just one. This might also increase the overlap size of the merged databases, which is important when identifying PKG-targets that

can be included in the simulation. Simulating endogenous signaling dynamics is a complicated endeavor. Many factors need to be included before the simulations can begin to meaningfully mimic the flow of molecular interactions that take place within organisms. For the simulation approach presented in the manuscript, it would most likely be an improvement to include a tissue-constraint, so the simulations would reflect the cell types that are being analyzed. However, identifying tissue-specific interactions is an ongoing unsolved issue, and presenting possible solutions to the problem is difficult.

It can be assumed that general ideal simulations of biological events would require certain stages: i) a balanced stage before "the event", ii) simulation of changes taking place upon execution of the "the event", iii) a new established balance after occurrence of "the event". The simulation approach presented in the manuscript only include steps similar to i) and ii). This means that potential counter-regulatory mechanisms that exists in biological systems are ignored. Including functionalities that enables a return to a step iii) would also make it possible to estimate a simulation-predicted decrease in NOX4 and NOX5 gene expression.

DeepCLIP and Scellnetor apply different methodologies. Obvious differences are their machine learning techniques: DeepCLIP utilizes supervised learning and Scellnetor utilizes unsupervised learning. Though both techniques can solve complicated tasks, they come with some challenges. A challenge of supervised learning is that prior knowledge is needed before data labelling is possible. DeepCLIP models were trained using CLIP data and random genomic backgrounds, as two differently labelled classes. As the background sets were drawn at random, they may include sequences that contain functional binding sites of the analyzed RBPs. Furthermore, the CLIP-sequences may include false positives, due to e.g. minor experimental imprecisions. Altogether, this may disturb the models' understanding of the RBPs' binding preferences. Finding a way to prevent the inclusion of false positives and false negatives in this scenario is a difficult task, but it may improve the performances of the DeepCLIP models.

Prior knowledge is not needed in unsupervised learning strategies. But these methods are challenged in other ways. For hierarchical clustering algorithms, and thereby with Scellnetor, it is a difficult task to find an optimal number of clusters. Scellnetor constructs a dendrogram until a user-defined threshold (number of clusters and minimum size) is reached and only shows users the clusters that fit the user-defined criteria. A possible improvement could be to implement the clustering-defining hyperparameters as a range. For example, if a researcher wants to find five clusters of size 5-15, the clustering should stop when five clusters of size 15 have been found, but the displayed clusters should include all clusters in the size-range 5-15. Another strategy could be to compute an "optimal" clustering using e.g. a squared error function. Such a computation would need the full dendrogram containing all genes in the analyzed data set and might as a consequence result in very large cluster-subnetworks that are difficult to inspect and analyze. Also, it may keep Scellnetor-users from exploring other possible solutions that are equally good.

The computational analysis presented in the third manuscript does not utilize machine learning techniques, but mathematical modeling to simulate effects of kinase activity. Computational simulations and machine learning both depend on algorithms and mathematical operations to produce outputs. But, where clustering methods and classifiers seek to partition data into groups or classes, simulations try to mimic biological phenomena quantitatively; making it possible to track the behavior of individual molecules included in the simulations [9]. A common denominator, however, is that the results of both depend on the quality of the input data. In general, simulations with many relevant details and high molecular resolution produce more meaningful outputs [10]. The simulation approach presented in chapter four, is relatively "sparse" as it does not include a high level of details as e.g. computations of molecular behavior based on thermodynamics. Thus, as already stated, it may benefit from an addition of additional layers of "biological rules" that co-drive the behavior of the simulated molecular entities.

In conclusion, this thesis introduced ongoing challenges that have arisen with the advent of the big data era. The three manuscripts included in this thesis presented novel bioinformatic tools that can analyze different types of biological data and discover new biological and biomedical insights. The tools have been implemented as freely available webtools and standalone programs, making it straightforward for researchers to readily use the programs to analyze big biological data. The manuscripts, thereby, served as examples of how "big data analytics in bioinformatics" might be approached in current research strategies.

5.1 References

1. McInnes L, Healy J, Melville J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv [stat.ML]. 2018. Available: http://arxiv.org/abs/1802.03426

2. Becht E, McInnes L, Healy J, Dutertre C-A, Kwok IWH, Ng LG, et al. Dimensionality reduction for visualizing single-cell data using UMAP. Nat Biotechnol. 2018. doi:10.1038/nbt.4314

3 Oughtred R, Stark C, Breitkreutz B-J, Rust J, Boucher L, Chang C, et al. The BioGRID interaction database: 2019 update. Nucleic Acids Res. 2019;47: D529–D541.

4 Larsen SJ, Schmidt HHHW, Baumbach J. De Novo and Supervised Endophenotyping Using Network-Guided Ensemble Learning. Systems Medicine. 2020;3: 8–21.

5. Zhang S, Zhou J, Hu H, Gong H, Chen L, Cheng C, et al. A deep learning framework for modeling structural features of RNA-binding protein targets. Nucleic Acids Res. 2016;44: e32.

6. Pan X, Rijnbeek P, Yan J, Shen H-B. Prediction of RNA-protein sequence and structure binding preferences using deep convolutional and recurrent neural networks. BMC Genomics. 2018;19: 511.

7. Alipanahi B, Delong A, Weirauch MT, Frey BJ. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. Nat Biotechnol. 2015;33: 831–838.

8. Maticzka D, Lange SJ, Costa F, Backofen R. GraphProt: modeling binding preferences of RNA-binding proteins. Genome Biol. 2014;15: R17.

9. Kaazempur-Mofrad MR, Bathe M, Karcher H, Younis HF, Seong HC, Shim EB, et al. Role of simulation in understanding biological systems. Comput Struct. 2003;81: 715–726. doi:10.1016/S0045-7949(02)00481-9

10. Feig M, Sugita Y. Whole-Cell Models and Simulations in Molecular Detail. Annu Rev Cell Dev Biol. 2019;35: 191–211. doi:10.1146/annurev-cellbio-100617-062542

6 Supplementary materials for Manuscript 1

Supplementary Method Section

Introduction to Scellnetor

Scellnetor is the first network-constraint time-series clustering algorithm implemented as interactive webtool to identify modules of genes connected in a molecular interaction network that show differentiating temporal expression patterns. Scellnetor allows unraveling network modules driving, for instance, cell differentiation or disease progression on a system biological level. In addition, the mechanisms identified by Scellnetor can be utilized for developing mechanistically oriented drugs that work on subnetwork-defined disease endophenotypes. Scellnetor is available as an intuitive and interactive online tool at https://exbio.wzw.tum.de/scellnetor and for download at GitLab (https://gitlab.com/AlexTheKing/scellnetor_docker_public).

Scellnetor requires Scanpy-generated ANNDATA (1) objects as raw data input and scanpy-generated plots as template-plots. The user can interactively select cells of interest for the network module identification (Fig. S6). In order to form a time-series, the cells are sorted according to pseudotime, since it contains information about differentiation statuses of cells (2) (sorting according to an alternative key available in the ANNDATA object is supported as well). One of Scellnetor's key features is the comparison of two distinct sets of single cells, which are either two paths through the template-plot or two sets of clusters, respectively. Scellnetor will compute a hyper-similarity matrix when comparing two sets. The hyper-similarity matrix contains information on how the genes are expressed compared to one another in the individual sets and compared to the genes in the set they are compared against. In case the user does not want to compare two sets of cells but look for the network module showing the most consistent expression profile, a standard distance matrix is calculated.

Scellentor uses a constrained hierarchical agglomerative clustering algorithm for extracting subnetworks of genes. Per default, the clustering is constrained by the interactions of the human PPI network from BioGrid (3), which makes it possible for Scellnetor to output clusters as connected subnetworks of genes. When comparing two paths or sets of clusters, the user can choose between searching for clusters of genes that are either differently or similarly expressed in the two sets. In addition to the connected subnetworks, Scellnetor outputs TSV files with statistically significant GO-terms for each cluster and plots of the mean and 95% confidence intervals of the expression patterns of the genes in the clusters.

The focus of this section will be on the functionalities associated with path-drawing and calculation of the hyper-similarity matrix, since these are the main novelties of the Scellnetor methodology. To quickly find out what a hyper-similarity means, see subsection "Summary - what do hyper-similarity values mean?". All examples in this section will assume that the user has drawn paths on a canvas-plot. From the sub-section "Extraction of cells and smoothing of timelines" till the end of this section, the data processing for sets of clusters and user-drawn path is the same.

Extraction and conversion of genes from uploaded file

As mentioned before, Scellnetor constrains the clustering by the interactions of the Biogrid network. Nevertheless, the user can also utilize any other network provided in a tab separated edge list. The only restriction is the required usage of human Entrez IDs as node identifiers. The same restriction applies to the genes of uploaded H5AD files, which also are required to be given as EntrezID (or as identifiers that can be converted to Entrez IDs) and are additionally available as nodes in the network. All IDs without a corresponding network node will be discarded from further analysis. Scellnetor automatically recognizes human Entrez IDs, human gene symbols, human Ensemble stable IDs and mouse gene

symbols. Due to overlapping gene symbol usage and identical human-mouse gene orthology of certain genes, some human and mouse gene symbols are mapped to the same human Entrez IDs. In these cases, Scellentor will automatically represent these Entrez IDs by the first instance of the genes in question that Scellnetor meets in its search. We recommend that users integrate a list with either human Entrez IDs or Ensemble stable IDs as an ANNDATA.var column, since this will preserve the highest number of genes that can be used for the clustering.

Selecting clusters on the canvas-plot

Users select cells on a canvas-plot, and the cluster to which the cell belongs is colored in accordance with the chosen set-color (Fig. S6). Switching between colors and selecting cells in different clusters makes it possible to create two distinct sets of clusters that can be compared.

Drawing paths on canvas-plot

When drawing paths with Scellnetor, users select cells on the canvas-plot, which will be automatically connected to a path. The order in which the cells are selected directs the construction of the path through the plot. Let Z be the set of all cells and $X = \{x_1, x_2, x_3 \dots x_n\}$ the set of user-selected cells. Then the path is created by forming a path between x_1 and x_2 followed by a path between x_2 and x_3 and so forth.. A path between cell x_i to cell x_{i+1} is generated by finding the cell, x_{ia}, that is closest to x_i and closer to x_{i+1}than x_i is. This is followed by finding the cell, x_{ib}, that is closest to x_{ia} and closer to x_{i+1}than x_{ia} is. This is repeated until a sub-path between x_i and x_{i+1} has been drawn. This is done for all possible pairs of x_i and x_{i+1} in X. The result is a minimal greedy path, which is the path connecting the cells in X with the fewest number of cells possible when following the rules of the greedy path algorithm. The cells in the resulting path are stored in the set, X_{path}.

Expanding trajectories

The paths can be expanded (i.e., made thicker to include more cells) by including a user-defined percentage of cells that neighbor cells in the minimal greedy path. The user can expand the paths to contain pct percentage of the overall dataset. This is achieved by greedily adding the closet cells to cells which are already in X_{path} until the desired number of cells is reached. Euclidean distance is used as distance metric and $X_{expand} = X_{path}$ when $i = 0$ for $X_{path} = \{x_1, x_2, x_3 \dots x_N\}$ where N is the number of cells in the minimal greedy path.

Finding similar-sized paths

For optimal comparisons of paths, it is recommended that users make paths that contain the same number of cells (see Extraction of cells and smoothing of timelines for an explanation). Scellnetor has integrated functionalities that allow users to even out the sizes of their drawn trajectories. If two paths have been drawn and one has been expanded by inclusion of the pct percent closest points, and the other is in its minimal greedy path form, then let $X_{expanded}$ be the set containing the cells of the expanded trajectory and Y_{path} be the set containing the cells of the path, which is in its minimal greedy path form. To even out the sizes of $X_{expanded}$ and Y_{path}, Y_{path} is first expanded by the inclusion of the pct percent closest cells for every $y_i \in Y_{path}$ (see Expanding trajectories). The result is the set $Y_{expanded}$. If $|Y_{expanded}| = |X_{expanded}|$, then the objective has been reached. If $|Y_{expanded}| < |X_{expanded}|$, a cell from $Y_{expanded}$ is chosen at random and its closest neighbor that is not already in $Y_{expanded}$, is added to the set. This is

repeated until $|Y_{expanded}| = |X_{expanded}|$. Adding cells like this will promote a uniform expansion of the trajectory. If $|Y_{expanded}| > |X_{expanded}|$, distances from the cells in Y_{path} to cells in $Y_{expanded} - Y_{path}$ are calculated. And the cells in $Y_{expanded} - Y_{path}$ that have the average longest distance to all cells in Y_{path} are removed until $|Y_{expanded}| = |X_{expanded}|$. Users can even-out their paths even if both paths have been expanded or none of them has. However, it is not possible to get a path smaller than the path's minimal greedy path form.

Extraction of cells and smoothing of timelines

A hyper-similarity matrix is only calculated when two sets are compared. From this subsection onwards till the subsection "Generation of results", it will be assumed that a user has drawn two paths A and B. Path A and path B are converted into matrices A and B, respectively. Matrix A has size $g \times m$ and B has size $g \times l$, where g is the number of genes, and m and l are the number of cells in path A and path B, respectively. Per default, a moving average function is applied on the rows of A and B, which results in new matrices called A_{mean} and B_{mean} with sizes $g \times m'$ and $g \times l'$, respectively. The element in row i and column j of A_{mean} and B_{mean} are found using the following moving average formula:

$$A_{mean_{i,j}} = \frac{1}{a}\sum_{z=0}^{a-1} A_{i,j+z} \quad (1)$$

where a is the user-defined size of the moving average and $A_{i,j}$ is the element in row i and column j of matrix A. Formula (1) is applied on every row of A and B and every j-th column in the range $\{j \in N \mid 0 \leq j \leq m - a + 1\}$ for A and every j-th column in the range $\{j \in N \mid 0 \leq j \leq l - a + 1\}$ for B. In cases where the two paths are of different lengths, the moving average window of the longer path is adjusted such that both paths contain the same number of smoothed values. This means, if a path contains a much higher number of cells than the path it is compared to, the moving average can "flatline" the gene expression patterns of the larger set. Therefore, it is ideal if the objects to be compared have identical sizes.

UMAP of gene expression of single cells

Before the gene expression patterns in the two sets are compared, the smoothed data in A_{mean} and B_{mean} are dimensionality reduced. UMAP (4) is applied for the dimensionality reduction given one of the four user-chosen distance metrics: Euclidean distance, Manhattan distance, Minkowski distance or Correlation. Scellnetor sets the UMAP parameters *n_neighbors*=30, *min_dist*=0.0 and *random_state*=42. The remaining UMAP parameters are set to their default values. A_{mean} and B_{mean} are concatenated before transformed by UMAP, as the gene expression patterns from the two matrices thereby will be high-dimensional coordinates on the same manifold structure. This will provide more meaningful inter-coordinate distances of the dimensionality reduced data. UMAP outputs a different coordinate landscape depending on the order of the concatenation. For example, the distances between coordinates resulting from a UMAP-embedding of the concatenation of A_{mean} and B_{mean} are similar, but slightly different from the distances between coordinates resulting from a UMAP-embedding of the concatenation of B_{mean} and A_{mean}.

So, to retain determinism of the Scellnetor clustering, a dimensionality reduction is conducted on both the concatenation of A_{mean} and B_{mean}, and the concatenation of B_{mean} and A_{mean}. The resulting dimensionality-reduced matrices, $C_{small_{AB}}$ and $C_{small_{BA}}$, both have the size $2g \times 2$, where g is the number of genes in A_{mean} and B_{mean}. $C_{small_{BA}}$ is redefined as the concatenation of $C_{small_{BA}}[g:]$ on top

of $C_{small_{BA}}[:g]$, as the order of the genes in the two coordinate sets should be identical for the further processing. $C_{small_{BA}}[g:]$ is the slice of $C_{small_{BA}}$ that contains the last g rows and $C_{small_{BA}}[:g]$ is the slice of $C_{small_{BA}}$ that contains the first g rows.

Hyper-similarity matrix

When comparing the two paths, path A and path B, a hyper-similarity matrix is computed. $C_{small_{AB}}[:g]$ contains the moving average-modified and dimensionality reduced gene expression patterns from path A, and $C_{small_{AB}}[g:]$ contains the moving average-modified and dimensionality reduced gene expression patterns from path B. The same holds true for $C_{small_{BA}}[:g]$ and $C_{small_{BA}}[g:]$, respectively. Again, g is the number of genes in A_{mean} and B_{mean}. A $C_{small_{AB}}$ vs. $C_{small_{AB}}$ distance matrix, $D_{small_{AB}}$, and a $C_{small_{BA}}$ vs. $C_{small_{BA}}$ distance matrix, $D_{small_{BA}}$, is calculated using Euclidean distance as metric. Again, to retain determinism of the Scellnetor clustering approach, a distance matrix, D_{small}, is found by

$$D_{small} = \frac{D_{small_{AB}} \oplus D_{small_{BA}}}{2}, \tag{2}$$

where the values of D_{small} are normalized by

$$D'_{small} = \frac{D_{small}}{max(D_{small})}. \tag{3}$$

In the matrix D_{small} in our example (Fig S4), the upper left quadrant corresponds to a path A vs. path A distance matrix and the lower right quadrant corresponds to a path B vs. path B distance matrix. The upper triangle of the upper right quadrant corresponds to the upper triangle of a path A vs. path B distance matrix and the lower triangle of the upper right quadrant corresponds to the lower triangle of a path B vs. path A distance matrix.

The diagonal of D_{small} is zeroed out. The diagonal of the upper right quadrant of D_{small} is zeroed out as well. the latter will be explained in the subsection "Weighting values of the hyper-similarity matrix" below. The upper triangular matrix of the upper left quadrant is defined as D_{AA}, the upper triangular matrix of the lower right quadrant is defined as D_{BB}, the upper triangular matrix of the upper right quadrant is defined as D_{AB} and the lower triangular matrix of the upper right quadrant is defined as D_{BA}. The values in D_{AA} and D_{BB} are reversed by

$$D'_{XX} = \left|\left(D_{XX} - (1 - 10^{-6})\right)\right| \tag{4}$$

where D_{XX} is the matrix and 10^{-6} is a small value subtracted from 1 to set the values of the matrix in the range $]0; 1]$. This way, no values are zeroed out after reversion of the distance-values and the relative distance-differences remain unchanged. If the user wants clusters of connected genes that have similar expression patterns in path A and path B, respectively, and similar expression patterns when comparing path A and path B, then D_{AB} and D_{BA} are updated using (9) (*cluster type 1*).

The matrices D_{AB} and D_{BA} remain as initially defined, if the resulting clusters should be connected genes that have similar expression patterns in path A and path B, respectively, and dissimilar expression patterns when comparing path A against path B (*cluster type 2*). To compress the distances from D_{AA}, D_{BB}, D_{AB} and D_{BA} into a single "pre-hyper-similarity matrix", D, the four matrices are element-wise multiplied as follows

$$D = D_{AA} \odot D_{BB} \odot D_{AB} \odot D_{BA}^{\top} \tag{5}$$

where $\odot$ it the Hadamard product. The diagonal and the lower triangular matrix of D are zeroed out. Now, the matrix D only needs a few processing steps before it is ready for the clustering as a hyper-

similarity matrix. The matrices D_{AA}, D_{BB}, D_{AB} and D_{BA} corresponds to the matrices D_{rr}, D_{gg}, D_{rg} and D_{gr}, respectively, on Figure S4.

Weighting values of the hyper-similarity matrix

In D, the values depend on the distances between the gene expression patterns from the cells in path A and on the distances between the gene expression patterns from the cells in path B. The matrices A_{mean} and B_{mean}, derived from the two paths, contain the moving average modified expression values of the same genes in the same order, but measured as the cells in the two sets follow different differentiation trajectories. The matrix element at $D_{i,j}$ depends on the expression values of gene i in A_{mean} and B_{mean}, respectively, and on the expression values of gene j in A_{mean} and B_{mean}, respectively. The matrix element at $D_{i,j}$ should also depend on the distance of gene i in A_{mean} vs. gene i in B_{mean} and gene j in A_{mean} vs. gene j in B_{mean}. Until now, these values have been zeroed out and ignored, but they will be used to weigh the matrix D in the following steps (Fig. S4).

For example, if a user applies Euclidean distance as metric and wants to find clusters of *cluster type 1* then the value in row i and column j of D, should be "penalized" if $A_{mean_{i,:}}$ and $B_{mean_{i,:}}$ are far away from each other in the Euclidean space and/or if $A_{mean_{j,:}}$ and $B_{mean_{j,:}}$ are far away from each other in the Euclidean space. $A_{mean_{i,:}}$ indicate the entire row i of A_{mean}. To get the distances of genes with identical IDs in A_{mean} and B_{mean}, new coordinate sets are defined: $C_{AB_A} = C_{small_{AB}}[:g]$, $C_{AB_B} = C_{small_{AB}}[g:]$, $C_{BA_A} = C_{small_{BA}}[:g]$ and $C_{BA_B} = C_{small_{BA}}[g:]$, where g is the number of genes in A_{mean} and B_{mean}. Two vectors, v_{AB} and v_{BA}, are calculated by finding the distances between every identically indexed row of C_{AB_A} and C_{AB_B} and every identically indexed row of C_{BA_A} and C_{BA_B}, respectively, such that $|v_{AB}| = |v_{BA}| = g$. A vector, v, is defined by

$$v = \frac{v_{AB} + v_{BA}}{2}. \tag{6}$$

If the aim is to find clusters of *cluster type 1*, then v is normalized as follows

$$v' = v - min(v), \tag{7}$$

$$v'' = \left|\left(\frac{v'}{(max(v') + 10^{-6})} - 1\right)\right|. \tag{8}$$

And if clusters of *cluster type 2* is the objective, then v is normalized by

$$v' = v - (min(v) - 10^{-6}), \tag{9}$$

$$v'' = \frac{v'}{max(v')}. \tag{10}$$

In both scenarios, v will be in the range $]0; 1]$, which implies that nothing will be zeroed out by the weighting of v. The hyper-similarity matrix is found by weighting row $D_{i,:}$ by element v_i and weighting of column $D_{:,j}$ by element v_j where i is in the range $\{i \in N \mid 1 \leq i \leq g\}$ and j is in the range $\{j \in N \mid 1 \leq j \leq g\}$. As a final step, D is normalized by

$$D' = \frac{D}{max(D)}, \tag{11}$$

such that all possible gene-gene hyper-similarities are in the range $]0; 1]$.

Summary - what do hyper-similarity values mean?

When comparing two sets of cells, Scellnetor computes a hyper-similarity matrix. The hyper-similarity matrix contains information on how the genes are expressed relative to each other within a set of cells as well as between the compared sets of cells. The main goal of our similarity function is to find genes whose expression patterns are highly similar and conserved within each cell set, but dissimilar between the cell sets (*cluster type 2*). To exemplify this, a high hyper-similarity value (close to 1) between two genes, $gene1$ and $gene2$, implies the following, if e.g. Euclidean distance is used as metric:

i) Both genes express pseudo-timelines in two user-drawn paths, path A and path B; $gene1_A$ and $gene2_A$ are the pseudo-timelines from path A and $gene1_B$ and $gene2_B$are the pseudo-timelines from path B.
ii) The genes $gene1_A$ and $gene2_A$are in close proximity to each other and $gene1_B$ and $gene2_B$are in close proximity to each other.
iii) The distance between $gene1_A$ and $gene2_B$ is large and the distance between $gene1_B$ and $gene2_A$ is large.
iv) The distance between $gene1_A$and $gene1_B$is large and the distance between $gene2_A$and $gene2_B$ is large (weighting by values in vector v, Fig. S4).

Constrained hierarchical agglomerative clustering

The constrained hierarchical agglomerative clustering iteratively clusters genes together pairwise in order of descending hyper-similarity until all items have been assigned to a cluster or the user-defined threshold has been reached. The user-defined threshold is defined via user-chosen parameters that defines the minimum cluster size and the minimum number of clusters. Per default, the clustering is constrained by the interactions of the PPI network from BioGrid. For the clustering, the constraint means, i) two single genes can only be fused into a cluster if they are neighbors in the network, ii) a single gene needs to neighbor a gene in a cluster before it can be added to the cluster, iii) when merging two clusters, they need to have at least one gene each that are neighbors in the graph. The possible connections of a cluster are equal to the sum of all connections of genes in that cluster.

Generating results

When Scellnetor has run a clustering, users can download the main results and all data that were produced in the Scellnetor pipeline. The main results are:

1. Clusters or connected components of genes as PDF files and two edgelists for every cluster in CSV file format where nodes are denoted as both human entrez IDs and human gene symbols
2. One plot per cluster of mean expression of the genes with 95% confidence interval in PDF file format. On the x-axis is “Moving-average-modified number of single cells”. The cells are arranged after the variable selected as sorting key. The y-axis shows “Normalized moving-average-modified gene expression”. It has been normalized such that the highest value of the concatenation of A_{mean} and B_{mean} is 1 and the smallest is 0. This normalization only serves visualization purposes and is done after the completion of the hyper-similarity calculation.
3. TSV files containing statistically significant GO-terms associated with the clusters. It is generated using GOA-tools (5), which uses Fisher’s exact test to calculate p-values and the Benjamini-Hochberg procedure to adjust p-values.

Changing the ANNDATA objects and removing noise

After the user has selected a Scanpy-generated plot as template-plot, he/she can choose to remove some of the cells from the ANNDATA object and construct a new one. This functionality has been implemented in case the user only wants to analyze certain groups of cells and perhaps remove noisy cell groups. Here, it makes sense to select a template-plot that is colored by cluster annotations. As mentioned, the clusters you choose will be removed from the data and a new ANNDATA object will be generated that only uses the data that remain. Based on user-defined parameters and user-interaction, pseudotime and novel plots will be generated and stored in the new ANNDATA object. The new ANNDATA object is now ready for further analysis by the functionalities in the Scellnetor pipeline.

Pre-processing of data for Scellnetor hematopoiesis study

Using the scRNA-seq data from the 19 clusters defined by (6) (GSE72857), we created an initial count matrix and made sure that we could reproduce their measured cluster-wise average gene expressions before we constructed an ANNDATA object, made cell maps and computed pseudotime. Note, that we could not identify 10 genes from their list with cluster-wise average gene expressions. These anonymous genes were omitted from our analysis. This gave us a total of 2730 single cells that expressed 3451 genes. We pre-processed the count matrix the same way as in the study by (7), with the exception of using the top 1726 ($3451/2 \approx 1726$) most variable genes instead of the top 1000.

Pre-processing of data for Scellentor exhausted CD8 T-cell study

Using the scRNA-seq data from the five clusters they identified in (8) (GSE137007) we generated an ANNDATA object. The data was already pre-processed using the Seurat package. Additionally, we computed a diffusion map and calculated pseudo time. The “start cell” for the pseudotime inference was the cell in the progenitor cluster (cluster 1, Fig. 3a) that was furthest away from all progenies.

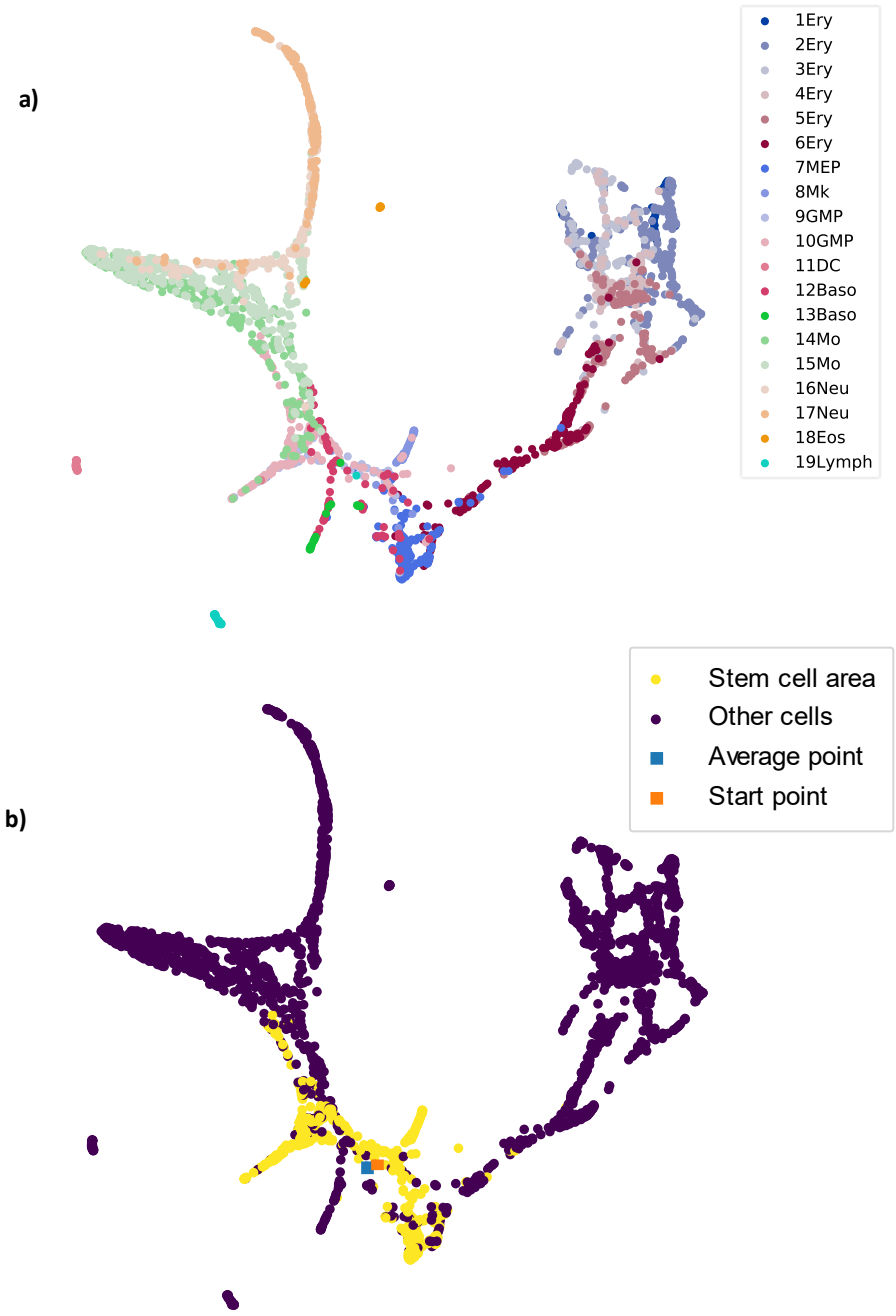


Figure S1. Stem cell area on cell map based on force-directed graph drawing of cell map. a Cell map shows localization of single cells from the study (6). On the right is shown the color codes for the cluster annotations. **b** Yellow (6) cells constitute the stem cell area comprised of cluster 7-10 from (6). Purple cells are all other cells. Blue point is the average position of the stem cell area (shown as a square). Orange point is the cell closest to the average position of the stem cell area and the "start cell" of the pseudotime computation (shown as a squares). On the right is shown a box with color coding of the cells.

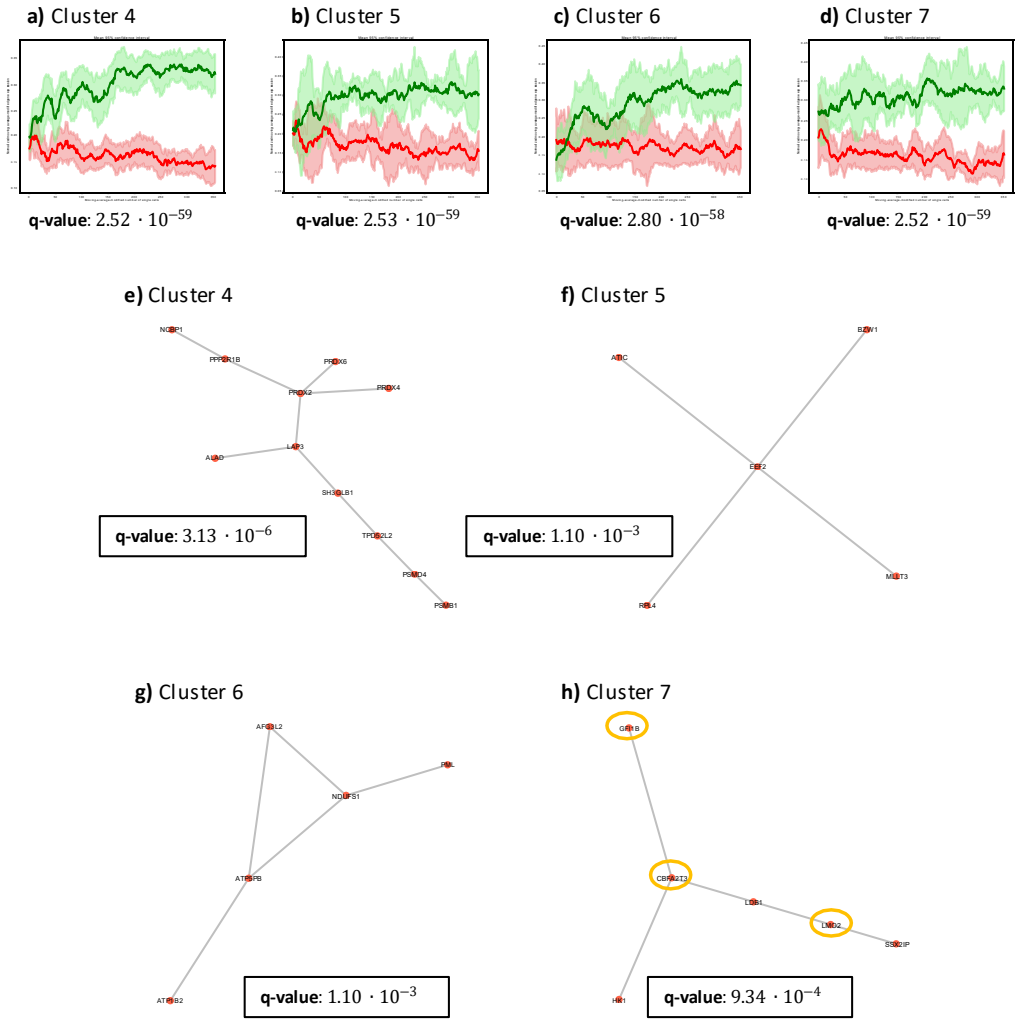


Figure S2. Clusters and mean pseudo-timelines and 95 % confidence intervals from Scellentor hematopoiesis study. a-d Gene modules and plots are based on cells from (6). Mean pseudo-timelines and 95 % confidence intervals of the genes in the Scellnetor clusters 4-7. The q-values below every plot are based on Wilcoxon signed-rank tests where the average expression over time of the clustered genes from the two distinct differentiation paths is compared. **e-h** Connected subnetworks of genes from clusters 4-7. The q-value is shown in the grey boxes near every cluster are based on a p-value from a Mann-Whitney *U* test, where all possible hyper-similarity values are compared with the values of the clusters. All p-values shown on the figure are corrected via the Benjamini-Hochberg Procedure to produce q-values.

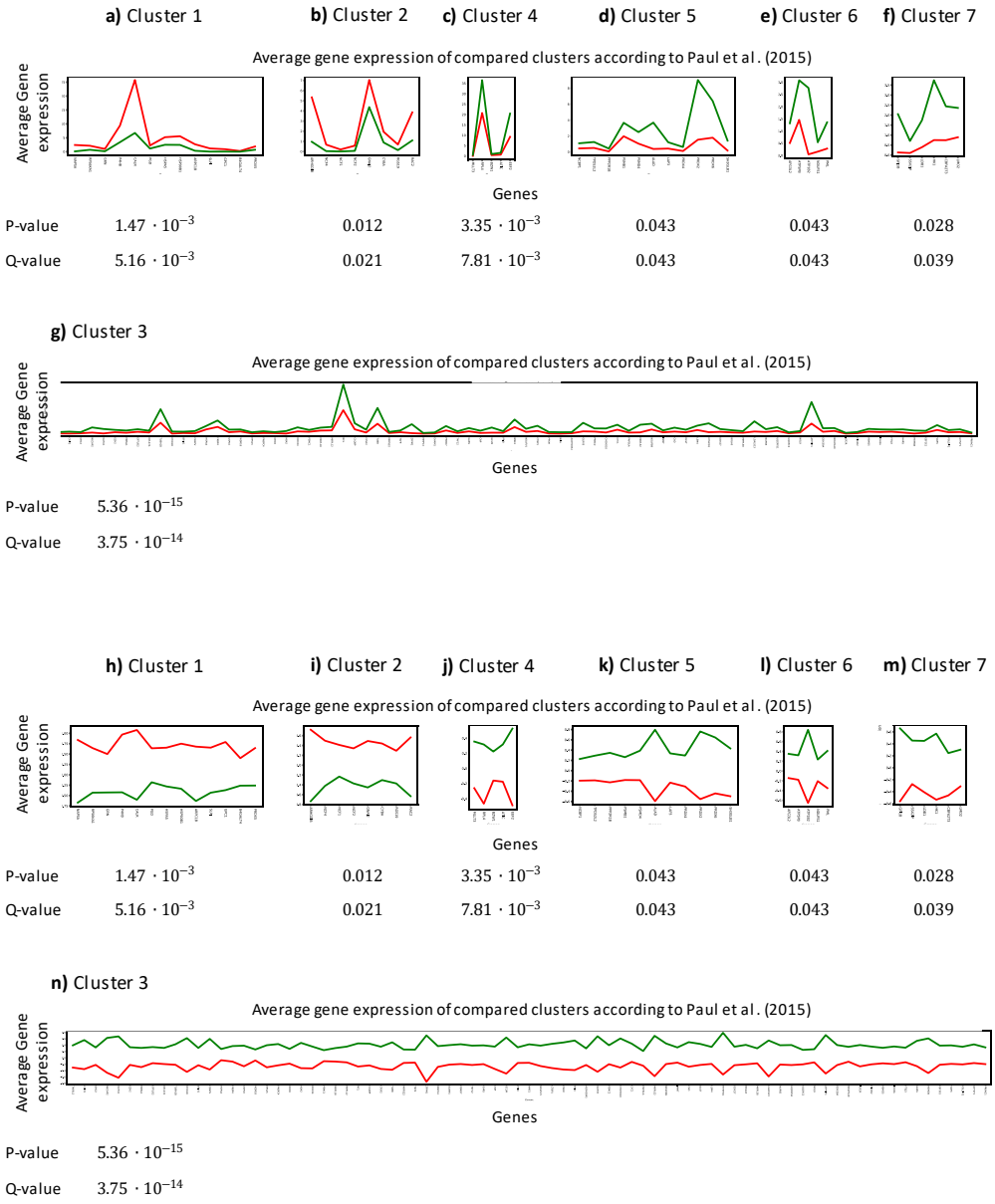
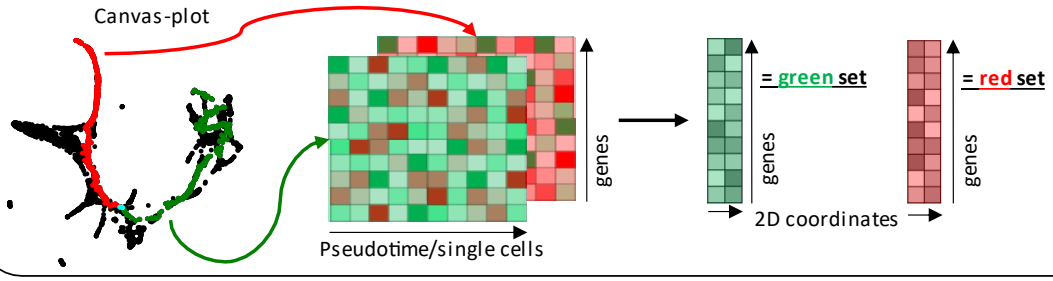


Figure S3. Average expression of genes in Scellnetor clusters. a-n Red lines indicate clusters 15-17 and green lines indicate clusters 1-6 from (6). **a-g** Correspond to Scellnetor hematopoiesis clusters 1-7. Along the first axes are names of genes in the Scellnetor hematopoiesis clusters. Along the second axes are shown the average gene counts. P-values and q-values are shown below every plot. **h-n** Red lines are based on the path through clusters 15-17 and green lines are based on the path through clusters 1-6 from (6). **h-n** Correspond to Scellnetor hematopoiesis clusters 1-7. Along the first axes are names of genes in the Scellnetor hematopoiesis clusters. Along the second axes are shown the average pre-processed gene expressions. P-values and q-values are shown below every plot. P-values are from Wilcoxon signed-rank tests where values from red lines are compared to values from green lines. Each p- and q-value pair are based on the values from the plots above them. Q-values are found using the Benjamini-Hochberg Procedure.

1. Extracting single cells and reducing dimensions of mRNA expression data

Extracting mRNA expression from single cells and sorting single cells according to pseudotime

Calculating 2D coordinates of expression data using UMAP

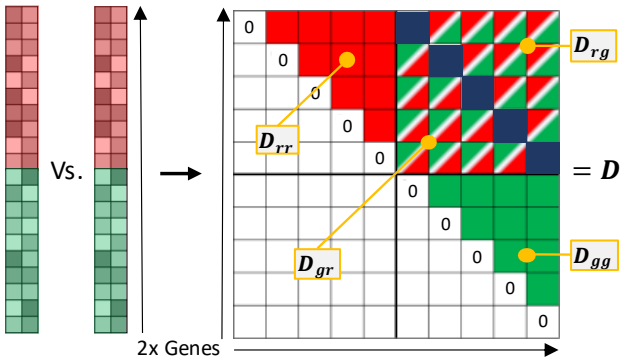


2. Finding distances of genes in single cell sets

Concatenating UMAP-representations of expression data and computing distance matrix, $\boldsymbol{D}$

Storing distances of identical genes from **red** and **green** sets in $\boldsymbol{v}$

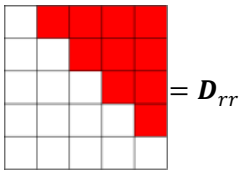
= Zero-out these values in distance matrix



$$\boldsymbol{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \dots \\ v_{|genes|} \end{bmatrix}$$

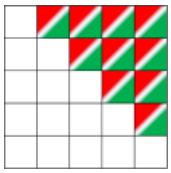
$|genes|$ = Number of genes in both **red** and **green** sets

red vs. **red** set



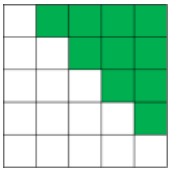
$= \boldsymbol{D_{rr}}$

red vs. **green** set



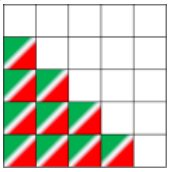
$= \boldsymbol{D_{rg}}$

green vs. **green** set



$= \boldsymbol{D_{gg}}$

green vs. **red** set



$= \boldsymbol{D_{gr}}$

= Zeros

3. Compute final hypersimilarity matrix

$$\boldsymbol{D}'_{rr} = \text{abs}\left(\frac{\boldsymbol{D}_{rr}}{\max(\boldsymbol{D})} - (1 - 10^{-6})\right)$$

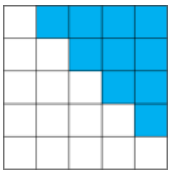
$$\boldsymbol{D}'_{gg} = \text{abs}\left(\frac{\boldsymbol{D}_{gg}}{\max(\boldsymbol{D})} - (1 - 10^{-6})\right)$$

$$\boldsymbol{D}'_{\text{rg}} = \frac{\boldsymbol{D}_{\boldsymbol{rg}}}{\max(\boldsymbol{D})}$$

$$\boldsymbol{D}'_{\text{gr}} = \frac{\boldsymbol{D}_{\boldsymbol{gr}}}{\max(\boldsymbol{D})}$$

$$\boldsymbol{D_{hyp}} = \boldsymbol{D_{rr}}\ \boldsymbol{D_{rg}}\ \boldsymbol{D_{gg}}\ \boldsymbol{D_{gr}^T}$$

$$\boldsymbol{D_{hyp}}' = \begin{array}{c} v_1\ v_2\ v_3\ v_{\dots}\ v_{|genes|} \\ *\ *\ *\ *\ * \end{array}$$



$*$ is multiplication of scalar and matrix row or column

Figure S4. Computation of hyper-similarity matrix - extended figure. **1** Two paths are drawn on the canvas-plot, red path and green path. Messenger-RNA expression data are extracted from selected cells and used to create expression matrices – one for each drawn path. The expression matrices are dimensionality reduced with UMAP, such that every pseudo-timeline of genes in the two sets are converted to 2D coordinates in the Euclidean space. **2** The 2D coordinates from the red set and the green set are concatenated along the first axis. The concatenated coordinate sets are used to produce a distance matrix, D, which is normalized by division of its max value. The distances between genes with identical IDs, but from different sets, are stored in the vector, v .The diagonal of the upper right quadrant of D is zeroed-out (top, right). Triangular matrices of the distance matrix' quadrant I, II and IV (as in cartesian coordinate system) are extracted. Matrix D_{rr} corresponds to a distance matrix of "red set vs. red set", matrix D_{gg} corresponds to a distance matrix of "green set vs. green set", matrix D_{rg} corresponds to a distance matrix of "red set vs. green set" and matrix D_{gr} corresponds to a distance matrix of "green set vs. red set". **3** Matrices D_{rr} and D_{gg} are modified such that genes with the shortest distances have the highest values and vice versa. The matrix D_{gr} is transposed and the Hadamard product of all triangular matrices is found (bottom, left). This produces the hyper-similarity matrix, D_{hyp}. The rows and columns of D_{hyp} are weighted by the values of v (bottom, right).

References

1. F. A. Wolf, P. Angerer, F. J. Theis, SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol.* **19**, 15 (2018).

2. S. Tritschler, *et al.*, Concepts and limitations for learning developmental trajectories from single cell genomics. *Development* **146** (2019).

3. R. Oughtred, *et al.*, The BioGRID interaction database: 2019 update. *Nucleic Acids Res.* **47**, D529–D541 (2019).

4. L. McInnes, J. Healy, N. Saul, L. Großberger, UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software* **3**, 861 (2018).

5. D. V. Klopfenstein, *et al.*, GOATOOLS: A Python library for Gene Ontology analyses. *Scientific Reports* **8** (2018).

6. F. Paul, *et al.*, Transcriptional Heterogeneity and Lineage Commitment in Myeloid Progenitors. *Cell* **164**, 325 (2016).

7. G. X. Y. Zheng, *et al.*, Massively parallel digital transcriptional profiling of single cells. *Nat. Commun.* **8**, 14049 (2017).

8. K. Kanev, *et al.*, Proliferation-competent Tcf1+ CD8 T cells in dysfunctional populations are CD4 T cell help independent. *Proc. Natl. Acad. Sci. U. S. A.* **116**, 20070–20076 (2019).

7 Supplementary materials for Manuscript 2

Table S1. This table shows the model parameters used when training DeepCLIP models on the curated dataset originally used by Maticzka et al, 2014.

Dataset	Epochs	Early stopping	CNN filter sizes (bp)	LSTM nodes	LSTM dropout	Output dropout
AGO1-4	200	20	4,5,6,7,8	10	0.1	0
AGO2	200	20	4,5,6,7,8	10	0.1	0
ALKBH5	500	50	4,5,6,7,8	10	0.1	0
C17ORF85	300	30	4,5,6,7,8	10	0.1	0
C22ORF28	200	20	4,5,6,7,8	10	0.1	0
CAPRIN1	200	20	4,5,6,7,8	10	0.1	0
ELAVL1	200	20	4,5,6,7,8	10	0.1	0
ELAVL1 (A)	200	20	4,5,6,7,8	10	0.1	0
ELAVL1 (B)	200	20	4,5,6,7,8	10	0.1	0
EWSR1	200	20	4,5,6,7,8	10	0.1	0
FUS	200	20	4,5,6,7,8	10	0.1	0
hnRNP C	200	20	4,5,6,7,8	10	0.1	0
HuR	50	5	4,5,6,7,8	10	0.1	0
IGF2BP1-3	200	20	4,5,6,7,8	10	0.1	0
MOV10	200	20	4,5,6,7,8	10	0.1	0
PTBP1	200	20	4,5,6,7,8	10	0.1	0
PUM2	200	20	4,5,6,7,8	10	0.1	0
QKI	100	10	4,5,6,7,8	10	0.1	0
SRSF1	200	20	4,5,6,7,8	10	0.1	0
TAF15	200	20	4,5,6,7,8	10	0.1	0
TDP43	50	5	4,5,6,7,8	10	0.1	0
TIA1	300	30	4,5,6,7,8	10	0.1	0
TIAL1	300	30	4,5,6,7,8	10	0.1	0
ZC3H7B	200	20	4,5,6,7,8	10	0.1	0

Table S2. This table shows performance metrics for the DeepCLIP models trained on the curated dataset originally used by Maticzka et al, 2014 (the table has been sliced into two for viewing puposes).

Measure	AGO1-4	AGO2	ALKBH5	C17ORF85	C22ORF28	CAPRIN1	ELAVL1	ELAVL1 (A)	ELAVL1 (B)	EWSR1	FUS	hnRNP C
CLIP method	PAR-CLIP	HITS-CLIP	PAR-CLIP	PAR-CLIP	PAR-CLIP	PAR-CLIP	PAR-CLIP	PAR-CLIP	PAR-CLIP	PAR-CLIP	PAR-CLIP	iCLIP
Number of positive binding sites	37402	48595	1347	2066	9869	8640	9095	9964	27775	16792	35081	21972
Accuracy	0.84317	0.77629	0.67079	0.82798	0.76287	0.88891	0.94200	0.93943	0.93968	0.92515	0.94978	0.94442
Positive Predictive Value	0.86109	0.79675	0.73641	0.87562	0.80401	0.87256	0.94597	0.94615	0.93255	0.94652	0.96382	0.94518
Negative Predictive Value	0.82288	0.75581	0.63304	0.79082	0.73022	0.90745	0.93801	0.93193	0.94720	0.90347	0.93518	0.94358
Sensitivity	0.84631	0.76555	0.53582	0.76553	0.70284	0.91447	0.93865	0.93954	0.94906	0.90864	0.93925	0.94807
Specificity	0.83948	0.78796	0.80677	0.89073	0.82440	0.86262	0.94540	0.93931	0.93012	0.94336	0.96132	0.94046
Precision	0.86109	0.79675	0.73641	0.87562	0.80401	0.87256	0.94597	0.94615	0.93255	0.94652	0.96382	0.94518
Recall	0.84631	0.76555	0.53582	0.76553	0.70284	0.91447	0.93865	0.93954	0.94906	0.90864	0.93925	0.94807
F1 score	0.85364	0.78084	0.62030	0.81689	0.75003	0.89302	0.94230	0.94283	0.94073	0.92719	0.95138	0.94662
Lift (at threshold)	159,348	153,054	146,732	174,700	158,847	172,088	187,484	177,986	184,868	180,453	184,246	181,809
Precision/Recall Break Even Point	0.85439	0.78409	0.66343	0.82136	0.76241	0.88808	0.94227	0.94274	0.93872	0.92638	0.95111	0.94624
Mean Average Precision	0.93184	0.87244	0.73008	0.90348	0.85384	0.94245	0.98261	0.98502	0.97828	0.97706	0.98735	0.98306
AUROC	0.91807	0.85891	0.71610	0.89843	0.83813	0.94789	0.98206	0.98155	0.98081	0.97349	0.98577	0.98274
AUROC area up to 50 negative samples	0.10041	0.04547	0.12447	0.31550	0.10980	0.16420	0.43492	0.37485	0.28804	0.28237	0.27713	0.26021
Rank of *last* (poorest ranked) positive case	68928	93216	2665	4031	19498	16603	19738	52194	18010	31860	66102	42079
Top 1: is the top ranked case positive	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000
Top 10: is there a positive in the top 10 ranked cases	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000
Slac Q-score	0.54873	0.38949	0.18053	0.51208	0.35341	0.66641	0.82149	0.81468	0.82007	0.77284	0.84575	0.82677
Mean Cross-Entropy	0.52482	0.67411	0.91371	0.59147	0.71303	0.41298	0.23799	0.24112	0.24593	0.29069	0.20541	0.23176
Root Mean Squared Error	0.33653	0.39112	0.46889	0.36046	0.40433	0.29085	0.21303	0.21589	0.21477	0.23975	0.19720	0.20911

Measure	HuR	IGF2BP1-3	MOV10	PTBP1	PUM2	QKI	SRSF1	TAF15	TDP43	TIA1	TIAL1	ZC3H7B
CLIP method	PAR-CLIP	PAR-CLIP	PAR-CLIP	HITS-CLIP	PAR-CLIP	PAR-CLIP	HITS-CLIP	PAR-CLIP	iCLIP	iCLIP	iCLIP	PAR-CLIP
Number of positive binding sites	125702	9039	14293	45074	9616	10776	19938	7798	92531	18549	42832	21462
Accuracy	0.98089	0.81869	0.87490	0.85331	0.92357	0.94204	0.88692	0.95641	0.82375	0.87857	0.87378	0.87420
Positive Predictive Value	0.98233	0.82673	0.87797	0.84744	0.95050	0.97863	0.88584	0.96521	0.84592	0.86735	0.87911	0.86653
Negative Predictive Value	0.97930	0.80814	0.87164	0.85955	0.89695	0.90678	0.88819	0.94705	0.79763	0.89242	0.86753	0.88270
Sensitivity	0.98118	0.84978	0.87908	0.86521	0.90114	0.91003	0.90291	0.95096	0.83118	0.90868	0.88615	0.89124
Specificity	0.98057	0.78035	0.87047	0.84118	0.94828	0.97780	0.86891	0.96239	0.81465	0.84498	0.85952	0.85636
Precision	0.98233	0.82673	0.87797	0.84744	0.95050	0.97863	0.88584	0.96521	0.84592	0.86735	0.87911	0.86653
Recall	0.98118	0.84978	0.87908	0.86521	0.90114	0.91003	0.90291	0.95096	0.83118	0.90868	0.88615	0.89124
F1 score	0.98175	0.83810	0.87852	0.85623	0.92516	0.94308	0.89429	0.95803	0.83848	0.88753	0.88262	0.87871
Lift (at threshold)	187,463	149,708	170,617	167,852	181,297	185,458	167,212	184,493	153,687	164,492	164,164	169,470
Precision/Recall Break Even Point	0.98172	0.83540	0.87817	0.85425	0.92164	0.93315	0.89172	0.95674	0.83936	0.88415	0.88181	0.87596
Mean Average Precision	0.99534	0.91020	0.93954	0.92437	0.97627	0.98230	0.95760	0.98416	0.92528	0.94640	0.94865	0.92655
AUROC	0.99500	0.89809	0.94021	0.92693	0.96887	0.97474	0.95511	0.98218	0.90454	0.94476	0.94334	0.93254
AUROC area up to 50 negative samples	0.21714	0.11602	0.11011	0.06402	0.53853	0.68199	0.19910	0.43603	0.06742	0.15028	0.09263	0.04420
Rank of *last* (poorest ranked) positive case	239231	15976	27249	88829	18247	20402	37190	14564	167904	35115	79709	41426
Top 1: is the top ranked case positive	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000
Top 10: is there a positive in the top 10 ranked cases	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000	100,000
Slac Q-score	0.93512	0.49364	0.62766	0.57442	0.76229	0.81332	0.67220	0.85873	0.50494	0.64081	0.63142	0.61217
Mean Cross-Entropy	0.10241	0.58174	0.44884	0.49808	0.30695	0.25777	0.38932	0.21008	0.56049	0.43281	0.43961	0.47274
Root Mean Squared Error	0.12817	0.35754	0.30616	0.32654	0.24588	0.21979	0.28719	0.19148	0.35196	0.30053	0.30415	0.31271

Table S3. This table shows the raw data used in Figure 1 and Figure S2. The highest AUROC score per dataset is indicated in bold.

Protein	GraphProt	iONMF	mDBN-	mDBN+	iDeepS	DeepCLIP
AGO1-4	0.895	0.892	0.872	0.881	0.89376	**0.91807**
AGO2	0.765	**0.921**	0.805	0.809	0.84595	0.85891
ALKBH5	0.68	**0.873**	0.686	0.714	0.63658	0.7161
C17ORF85	0.8	0.862	0.817	0.82	0.76033	**0.89843**
C22ORF28	0.751	**0.912**	0.783	0.792	0.82857	0.83813
CAPRIN1	0.855	0.928	0.825	0.834	0.93974	**0.94789**
ELAVL1	0.955	0.963	0.964	0.966	0.98088	**0.98081**
ELAVL1 (A	0.959	0.933	0.965	0.966	0.98118	**0.98206**
ELAVL1 (B	0.935	0.972	0.956	0.961	0.97929	**0.98155**
EWSR1	0.935	0.871	0.964	0.966	0.96971	**0.97349**
FUS	0.968	0.827	0.979	0.98	0.9832	**0.98577**
hnRNP C	0.952	0.959	0.961	**0.994**	0.97865	0.98274
HuR	0.991	0.958	0.994	0.994	0.99352	**0.995**
IGF2BP1-3	0.889	**0.928**	0.872	0.879	0.8898	0.89809
MOV10	0.863	0.939	0.831	0.854	0.93375	**0.94021**
PTBP1	0.937	0.86	0.879	**0.983**	0.92398	0.92693
PUM2	0.954	0.942	0.965	0.971	**0.97489**	0.96887
QKI	0.957	0.88	0.981	**0.983**	0.98194	0.97474
SRSF1	0.898	0.857	0.927	0.931	0.9489	**0.9551**
TAF15	0.97	0.927	0.98	**0.983**	0.98244	0.98218
TDP-43	0.874	0.855	0.874	**0.983**	0.90289	0.90454
TIA1	0.861	0.937	0.888	0.891	0.93527	**0.94476**
TIAL1	0.833	0.915	0.867	0.87	0.93809	**0.94334**
ZC3H7B	0.82	0.849	0.786	0.796	0.92907	**0.93254**
Average	0.8874	0.9067	0.8925	0.9084	0.9213	**0.934594**

Table S4. This table shows analysis results of DeepCLIP, GraphProt, and iDeepS predictions on eCLIP datasets from ENCODE and POSTAR2. The highest AUROC score for each dataset is indicated in bold. This table contains the raw data for Figure S5a-b.

Dataset	Source	DeepCLIP_auroc	DeepCLIP_loss	GraphProt	iDeepS
EWSR1_K562	ENCODE	0.47417	**0.48059**	0.28892	0.47658
FUS_HepG2	ENCODE	0.50563	**0.50666**	0.34830	0.49977
FUS_K562	ENCODE	0.48345	**0.49164**	0.36339	0.47652
HNRNPC_HepG2	ENCODE	0.47039	**0.49217**	0.36504	0.44132
HNRNPC_K562	ENCODE	0.51653	**0.54308**	0.38056	0.46065
IGF2BP1_HepG2	ENCODE	0.4434	**0.46435**	0.43257	0.40742
IGF2BP1_K562	ENCODE	0.58605	**0.59116**	0.52205	0.58209
IGF2BP2_K562	ENCODE	0.49822	**0.51751**	0.43837	0.45410
IGF2BP3_HepG2	ENCODE	0.43913	**0.46215**	0.40623	0.38314
PTBP1_HepG2	ENCODE	**0.75282**	0.74548	0.69209	0.75000
PTBP1_K562	ENCODE	0.77531	0.76930	0.75445	**0.78045**
PUM2_K562	ENCODE	0.63616	**0.64864**	0.64101	0.60928
QKI_HepG2	ENCODE	**0.66737**	0.64997	0.54964	0.59190
QKI_K562	ENCODE	**0.7301**	0.70635	0.64101	0.64102
SRSF1_HepG2	ENCODE	0.79586	0.78621	**0.87254**	0.80408
SRSF1_K562	ENCODE	0.75299	0.74631	**0.82510**	0.74996
TARDBP_K562	ENCODE	0.7452	0.74117	**0.76556**	0.74212
TIA1_HepG2	ENCODE	0.45259	0.46063	**0.50427**	0.43202
TIA1_K562	ENCODE	0.457	0.46066	**0.52276**	0.43533
TIAL1_HepG2	ENCODE	0.49793	**0.50129**	0.48484	0.49745
TAF15_HepG2_74nt	ENCODE	0.5234	0.54234	0.29299	**0.56343**
TAF15_K562_74nt	ENCODE	0.52418	**0.53571**	0.36274	0.5367
PTBP1	POSTAR2	0.85268	0.84535	0.81046	**0.85718**
QKI	POSTAR2	**0.75262**	0.73060	0.60129	0.57512
IGF2BP1	POSTAR2	0.6047	**0.61138**	0.52764	0.59793
EWSR1	POSTAR2	0.47549	**0.47922**	0.31755	0.47266
PUM2	POSTAR2	0.64414	**0.65647**	0.55256	0.59538
TIA1	POSTAR2	0.51946	0.52365	**0.63544**	0.51252
TARDBP	POSTAR2	0.91403	0.91482	0.88937	**0.91579**
HNRNPC	POSTAR2	0.46689	**0.50970**	0.30076	0.33990
IGF2BP2	POSTAR2	0.57719	**0.59459**	0.47847	0.54278
SRSF1	POSTAR2	0.81861	0.81657	**0.86164**	0.80181
IGF2BP3	POSTAR2	0.63975	**0.65126**	0.49924	0.60645
TAF15_74nt	POSTAR2	0.5623	0.56386	0.25642	**0.56938**

Table S5. This table shows analysis results of DeepCLIP models trained on RNAcompete data on the corresponding curated datasets from Maticzka et al, 2014, along with previous results of RNAcontext, RCK, DeepBind, DLPRB-CNN and DLPRB-RNN. The highest AUROC score for each dataset is indicated in bold. This table contains the raw data for Figure S5c.

Protein	RNAcontext	RCK	DeepBind	DLPRB-CNN	DLPRB-RNN	DeepCLIP_auroc	DeepCLIP_loss
FUS	0.7264	0.4512	0.25456	0.559330459	0.631797243	0.7582	**0.77412**
HNRNPC	**0.9536**	0.9504	0.94094	0.938988437	0.907115059	0.94559	0.94674
HUR	0.7209	0.808	0.90695	0.840627205	0.633953485	**0.92186**	0.91159
HUR	**0.8613**	0.7604	0.84578	0.826327769	0.834905447	0.81252	0.81801
HUR	0.7073	0.7668	0.75877	0.677171376	0.754192144	0.78712	**0.79976**
HUR	0.8126	0.7956	0.73366	0.89515894	0.922665792	**0.93469**	0.93423
HUR	0.9029	0.8628	0.89512	0.884332556	0.894433603	**0.91972**	0.91769
ELAV	0.8009	0.5934	0.78248	0.724345701	0.722000817	0.83845	**0.83858**
ELAV	0.7837	0.6056	0.76166	0.718269331	0.683028352	0.81957	**0.82039**
ELAV	0.7887	0.6463	0.75677	0.71152231	0.713995922	**0.81733**	0.81715
IGF2BP123	0.6705	**0.6747**	0.66079	0.650788131	0.663739922	0.66138	0.67138
IGF2BP123	0.7007	0.6853	0.68958	0.684364542	0.674215515	**0.72659**	0.69312
PTB1	0.8125	**0.8385**	0.83771	0.823884652	0.816149369	0.81779	0.79672
PTB1	0.3022	**0.8378**	0.82758	0.82367687	0.798384264	0.81223	0.80224
PUM	0.8415	0.7969	0.72109	0.822911027	0.436137882	**0.8576**	0.85325
PUM	0.5726	**0.8409**	0.81181	0.769568563	0.767893503	0.74776	0.75429
PUM	**0.8239**	0.8024	0.82054	0.807955895	0.524436589	0.80649	0.80154
PUM	0.7933	0.8025	0.80972	**0.809755159**	0.470014253	0.76795	0.74113
PUM	0.8152	0.8396	0.82001	**0.88577737**	0.878187407	0.85316	0.84723
PUM	0.8337	0.7611	**0.9013**	0.712784083	0.17793193	0.81228	0.79967
QKI	0.7259	0.8473	**0.92237**	0.815683951	0.846550434	0.28036	0.22752
SRSF1	0.7759	0.7863	0.45962	0.793122017	0.787141773	0.80231	**0.82405**
TIA1	**0.8496**	0.8329	0.67562	0.8304637	0.778402147	0.83343	0.82952

Table S6. This table shows DeepCLIP analysis results of exonic point mutations taken from the dataset previously published by Raponi et al, 2011. During preparation of this manuscript we uncovered some errors in the original dataset, which we corrected. This table contains the raw data for Figure 2c-d.

Due to the size of the table, it cannot be included in this thesis. Find the table here.

Table S7. This table shows DeepCLIP, GraphProt, and iDeepS analysis results of the exonic variants introduced in ACADM exon 5. This table contains the raw data for Figure 3b-c and Figure S36c-d.

ACADM exon 5 variant	PSI	hnRNP A1	SRSF1	Method
WT	0.87	0.48949665	0.727192342	DeepCLIP
c.361C	0.95	0.3477648	0.751877189	DeepCLIP
c.361G	0.94	0.48648754	0.680297613	DeepCLIP
c.361T	0.75	0.54268605	0.669966698	DeepCLIP
c.362A	0.31	0.5764722	0.688741028	DeepCLIP
c.362G	0.25	0.63222045	0.656009555	DeepCLIP
c.362T	0.08	0.5879869	0.668184936	DeepCLIP
c.363A	0.3	0.6961604	0.698751807	DeepCLIP
c.363C	0.92	0.38456744	0.762667596	DeepCLIP
c.363G	0.96	0.33724982	0.800012827	DeepCLIP
WT	0.87	0.29453328	0.7739826	iDeepS
c.361C	0.95	0.38432112	0.6904154	iDeepS
c.361G	0.94	0.31281966	0.76323426	iDeepS
c.361T	0.75	0.34277448	0.71046454	iDeepS
c.362A	0.31	0.5110051	0.75204766	iDeepS
c.362G	0.25	0.50457734	0.72424805	iDeepS
c.362T	0.08	0.3858132	0.71060866	iDeepS
c.363A	0.3	0.33466733	0.7736497	iDeepS
c.363C	0.92	0.25661853	0.8795054	iDeepS
c.363G	0.96	0.2292279	0.8596103	iDeepS
WT	0.87	0.0230146	-0.90378	GraphProt
c.361C	0.95	-0.02534	-0.994917	GraphProt
c.361G	0.94	0.0122239	-0.944936	GraphProt
c.361T	0.75	-0.0438069	-1.27013	GraphProt
c.362A	0.31	0.0353308	-1.23056	GraphProt
c.362G	0.25	0.0795215	-1.14543	GraphProt
c.362T	0.08	0.0600023	-1.24585	GraphProt
c.363A	0.3	0.0507217	-0.972072	GraphProt
c.363C	0.92	-0.00628	-0.747361	GraphProt
c.363G	0.96	0.0184861	-0.841334	GraphProt

Table S8. This table contains information about the oligos used for SPRi and the model estimates produced by the software applications CLAMP and scrubber, as well as the DeepCLIP prediction scores. This table contains the raw data for Figure 5c-d and Figure S44.

		SRSF1					
Maximum (model fit)	DeepCLIP	CLAMP Ka	CLAMP Kd	SRSF1 KD (M)	Rmax (scrubber)	Maximum (model fit)	DeepCLIP
49.55	0.11488	2.30E+01	1.40E-03	6.09E-05	71.60	28.32	0.48961
60.45	0.24764	3.82E+01	4.00E-04	1.05E-05	150.00	54.34	0.32697
130.64	0.34651	5.30E+02	4.83E-04	9.13E-07	342.00	111.63	0.78964
718.05	0.63135	5.44E+01	2.77E-04	5.09E-06	139.50	30.07	0.42171
1517.51	0.56735	2.09E+03	3.51E-04	1.68E-07	456.65	194.52	0.80300
1468.65	0.53705	15.78	0.00	2.48E-04	29.50	8.86	0.16036
612.41	0.31431	8.90E+02	5.16E-03	5.80E-06	79.00	38.13	0.39719
1149.35	0.43079	1.02E+01	1.65E-03	1.61E-04	115.00	20.96	0.29787
59.37	0.28019	NA	NA	NA	0	0	0.76529
71.55	0.28303	NA	NA	NA	0	0	0.36452
885.16	0.50425	1.61E+03	4.86E-04	3.02E-07	289.50	110.43	0.37348
1259.88	0.66315	9.10E+00	2.26E-03	2.49E-04	26.50	10.59	0.21561

Table S9. This table contains DeepCLIP predictions of the two SSOs used to correct splicing of *ACADM* exon 6.

Protein	Model id	SSO	Sequence	DeepCLIP prediction score
ALKBH5	ALKBH5	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.695646286
AGO2	AGO2	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.6800313
ALKBH5	ALKBH5	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.564591348
hnRNPA1	hnRNPA1	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.552014828
TDP-43	TDP43-2	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.518021643
AGO2	AGO2	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.516236007
TIAL1	TIAL1	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.489429832
TDP-43	TDP43	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.471341074
hnRNPA1	hnRNPA1	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.432616383
TDP-43	TDP43-2	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.408348113
SRSF1	SRSF1-2	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.388179183
TIA1	TIA1	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.37057963
C22ORF28	C22ORF28	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.365873963
TIAL1	TIAL1	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.356476128
SRSF1	SRSF1-2	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.345500499
hnRNPA1	hnRNPA1	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.332418829
C22ORF28	C22ORF28	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.329255998
AGO1-4	AGO1-4	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.316258371
AGO1-4	AGO1-4	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.279748529
hnRNPA1	hnRNPA1	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.272068977
IGF2BP1-3	IGF2BP1-3	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.206803709
TDP-43	TDP43	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.192072108
TIA1	TIA1	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.178311661
PUM2	PUM2	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.128042653
C17ORF85	C17ORF85	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.104126602
MOV10	MOV10	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.094622806
IGF2BP1-3	IGF2BP1-3	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.092992872
PUM2	PUM2	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.086418718
ZC3H7B	ZC3H7B	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.075361833
SRSF1	SRSF1	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.061138362
QKI	QKI	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.043321431
hnRNPC	hnRNPC	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.038740832
ZC3H7B	ZC3H7B	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.038646877
CAPRIN1	CAPRIN1	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.028823696
hnRNPC	hnRNPC	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.024413832
ELAVL1 (A)	ELAVL1A	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.023800772
C17ORF85	C17ORF85	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.023452237
EWSR1	EWSR1	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.022309046
ELAVL1	ELAVL1	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.02190683
QKI	QKI	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.020946642
ELAVL1	ELAVL1	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.020327197
MOV10	MOV10	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.019962186
EWSR1	EWSR1	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.017888084
ELAVL1 (A)	ELAVL1A	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.015540854
CAPRIN1	CAPRIN1	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.013676633
FUS	FUS	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.006430701
SRSF1	SRSF1	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.005627556
FUS	FUS	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.003811069
ELAVL1 (B)	ELAVL1B	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.00380164
ELAVL1 (B)	ELAVL1B	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.003207838
TAF15	TAF15	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.002246743
PTBP1	PTBP1	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.002214356
PTBP1	PTBP1	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.002202718
HuR	HuR	SSO2	UAAGUGUGAAAUAAAGCGGCAGUUA	0.002150088
TAF15	TAF15	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.001895699
HuR	HuR	SSO1	AGUGUGAAAUAAAGCGGCAGUUACA	0.001484048

PREVIOUSLY CURATED DATASET

The curated CLIP-seq dataset used for training models and measuring the Area Under Receiver Operator Curve (AUROC) are identical to the CLIP-seq datasets used in the GraphProt paper (Maticzka et al. 2014), with the difference that all extra padding was removed, retaining only the peak area for DeepCLIP training. The complete datasets containing foreground and background sets for training and validation were downloaded from http://www.bioinf.uni-freiburg.de/Software/GraphProt. According to (Maticzka et al. 2014), the individual CLIP-seq datasets are obtained from the doRiNA database (http://dorina.mdc-berlin.de). Sequences in datasets are based on hg19.

Datasets from the doRiNA database:

- Ago2 HITS-CLIP (Kishore et al., 2011)
- ELAVL1 PAR-CLIP(A) & HITS-CLIP (Kishore et al., 2011)
- ELAVL1 PAR-CLIP(B) (Lebedeva et al., 2011)
- ELAVL1 PAR-CLIP(C) (Mukherjee et al., 2011)
- HNRNPC iCLIP (Konig et al., 2010)
- MOV10 PAR-CLIP (Sievers et al., 2012)
- SFRS1 CLIP-seq (Sanford et al., 2009)
- TDP-43 iCLIP (Tollervey et al., 2011)
- TIA1 & TIAL1 iCLIP (Wang et al., 2010)
- EWSR1, FUS & TAF15PAR-CLIP (Hoell et al., 2011)
- Ago1-4, IGF2BP1-3, PUM2 & QKI PAR-CLIP (Hafner et al., 2010)
- ALKBH5, C17ORF85, C22ORF28, CAPRIN1, ZC3H7B PAR-CLIP (Baltz et al., 2012)

Dataset not from the doRiNA database:

- PTB HITS-CLIP (Xue et al., 2009), (GSE19323)

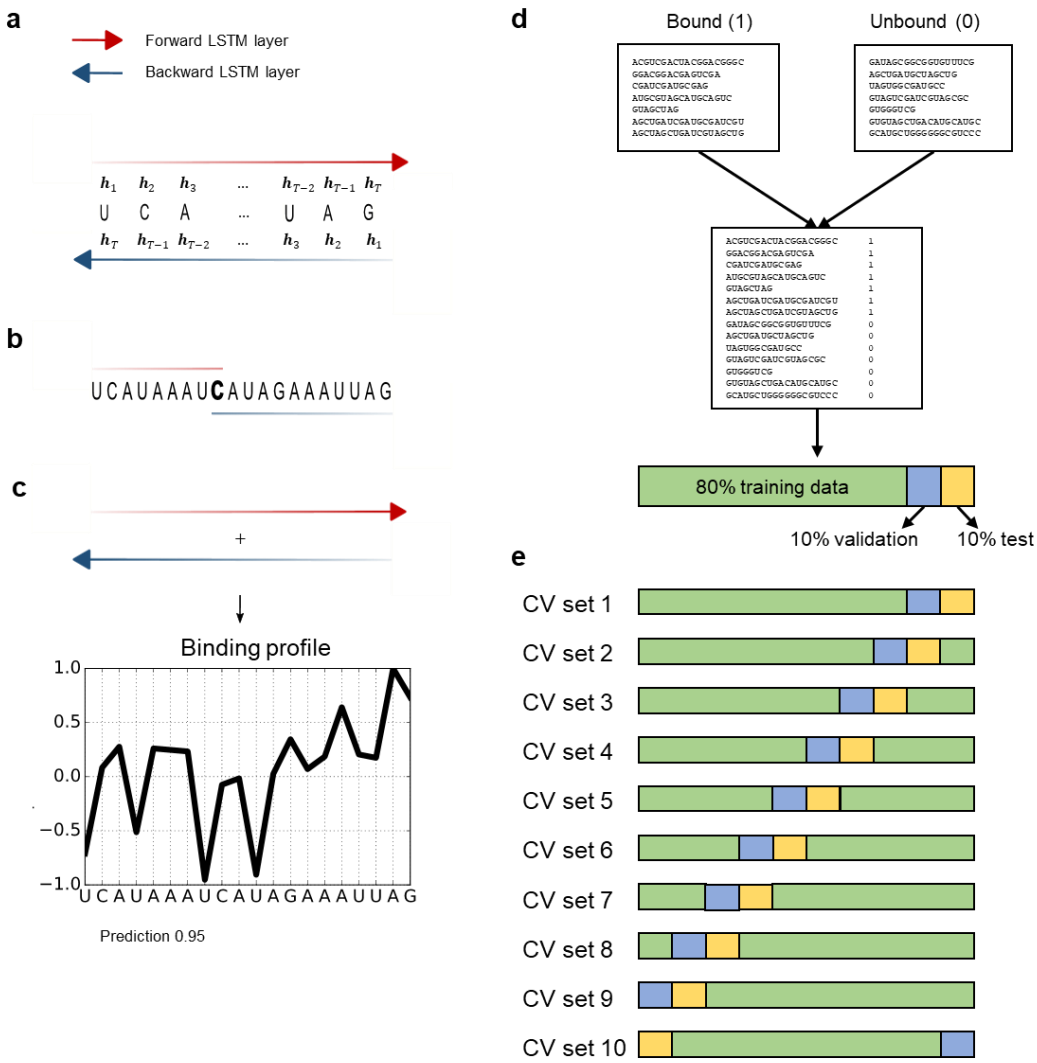


Figure S1 | Conversion of the BLSTM output to binding profile and DeepCLIP training. (a) The output of the forward LSTM layer is shown as a red arrow with a color intensity that increases gradually with the time-step number. The higher the color intensity and the higher the time-step number, the more contextual information has been available to the LSTM layer. The same is shown for the backward LSTM layer (blue arrow). The hidden state produced at the last time-step, , could contain information about the entire sequence. The outputs from the forward and backward LSTM layers are concatenated so hidden-states that are based on the base are combined. This figure is equal to from the BLSTM layer. Here, the time-step order is equal to the time-step order of the forward LSTM layer. (b) The contextual information that may present in BLSTM hidden-state that represents the highlighted cytosine (C in bold) is shown. The cytosine contains information about all the bases that surrounds it. (c) The binding profile is found via a summation of the vector elements in the BLSTM hidden-states. This results in a new vector that has a length equal to the input sequence. Every element of the vector indicates a class; positive value = class 1 (protein) and negative value = class 0 (genomic background). Every sequence has been divided by the vector element that has the highest absolute value. This results in a sequence that ranges from -1 to 1 and where it is easy to locate areas that are import for protein binding and areas that look like random genomic background. At the bottom, the prediction of the sequences in shown. (d) DeepCLIP is trained on input sequences belonging to either the bound class (assigned a score of 1) or the unbound (background) class (assigned a score of 0). The combined dataset is then divided into a training set, a validation set, and a test set. With default options they constitute 80%, 10%, and 10% of the dataset respectively.(e) When running in 10-fold cross-validation (CV) mode, the training set is divided into 10 different segmentations with a non-overlapping distribution of sequences in the validation and test set. These 10 different segmentations are then used to train 10 different models.

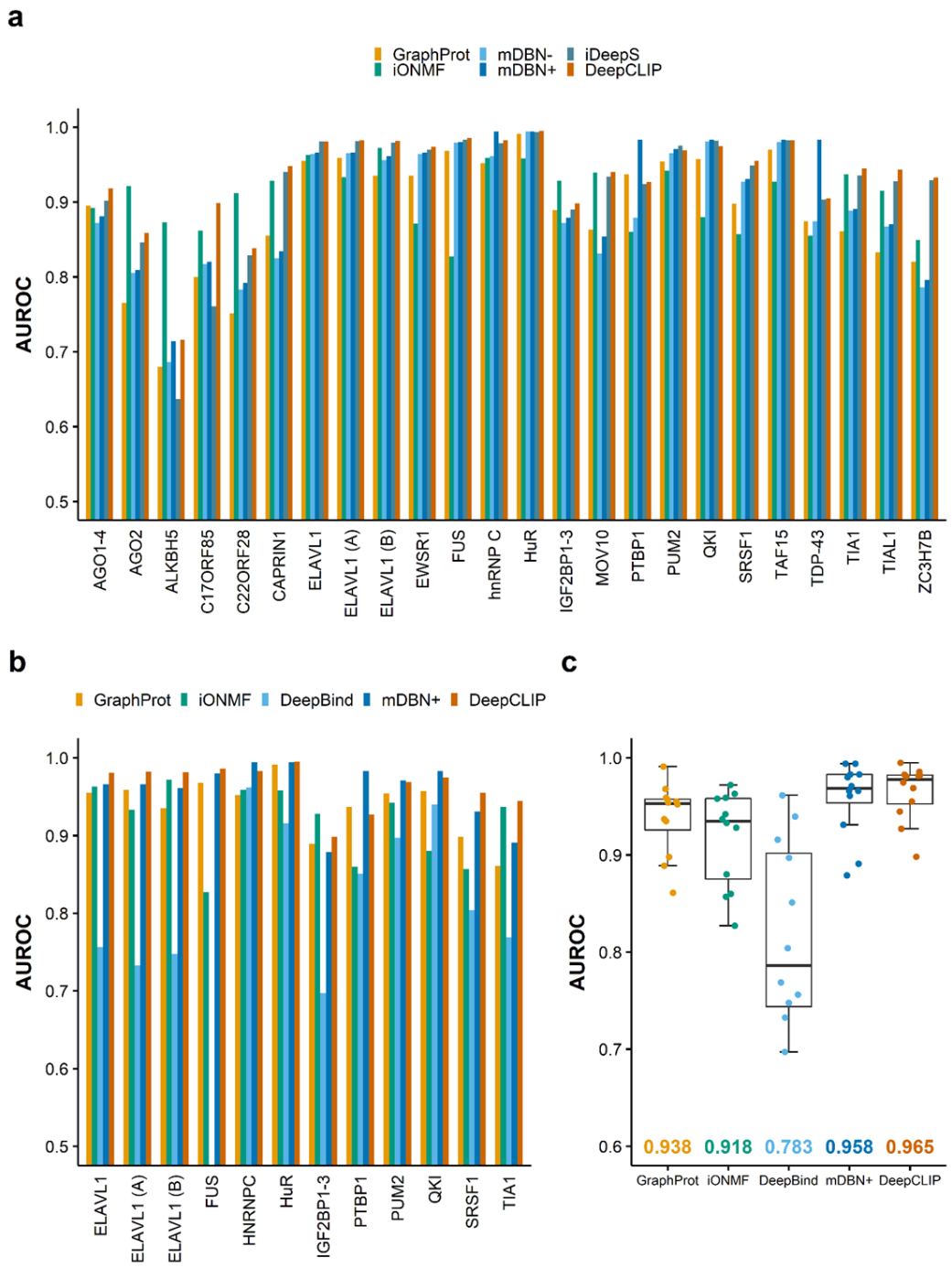


Figure S2 | Comparison of DeepCLIP AUROC performance on a benchmark dataset with existing tools. (a) Peak-sequences from the same input sequences used in Maticzka et al., 2014 were used to train DeepCLIP and iDeepS models using 10-fold cross-validation. Area under receiver operating characteristics (AUROC) for each protein were calculated using the combined predictions of all 10 models. AUROC for other methods were based on the reported scores from these studies. mDBN- indicates Deepnet with secondary structures, mDBN+ indicates Deepnet with secondary and tertiary structures. (b) AUROC metrics for the 12 proteins with DeepBind available were obtained by running DeepBind on the full dataset with all available models for the protein and using the model with the highest performance. All other values are identical to those in (a). Because deepnet with tertiary structures performed better than with just secondary structures, we show only this variant in this plot. (c) Boxplot of AUROC scores from each method for the 12 datasets. DeepBind produced an AUROC score of 0.31 for FUS, this datapoint is outside the plotting areas of both (b) and (c) plots.

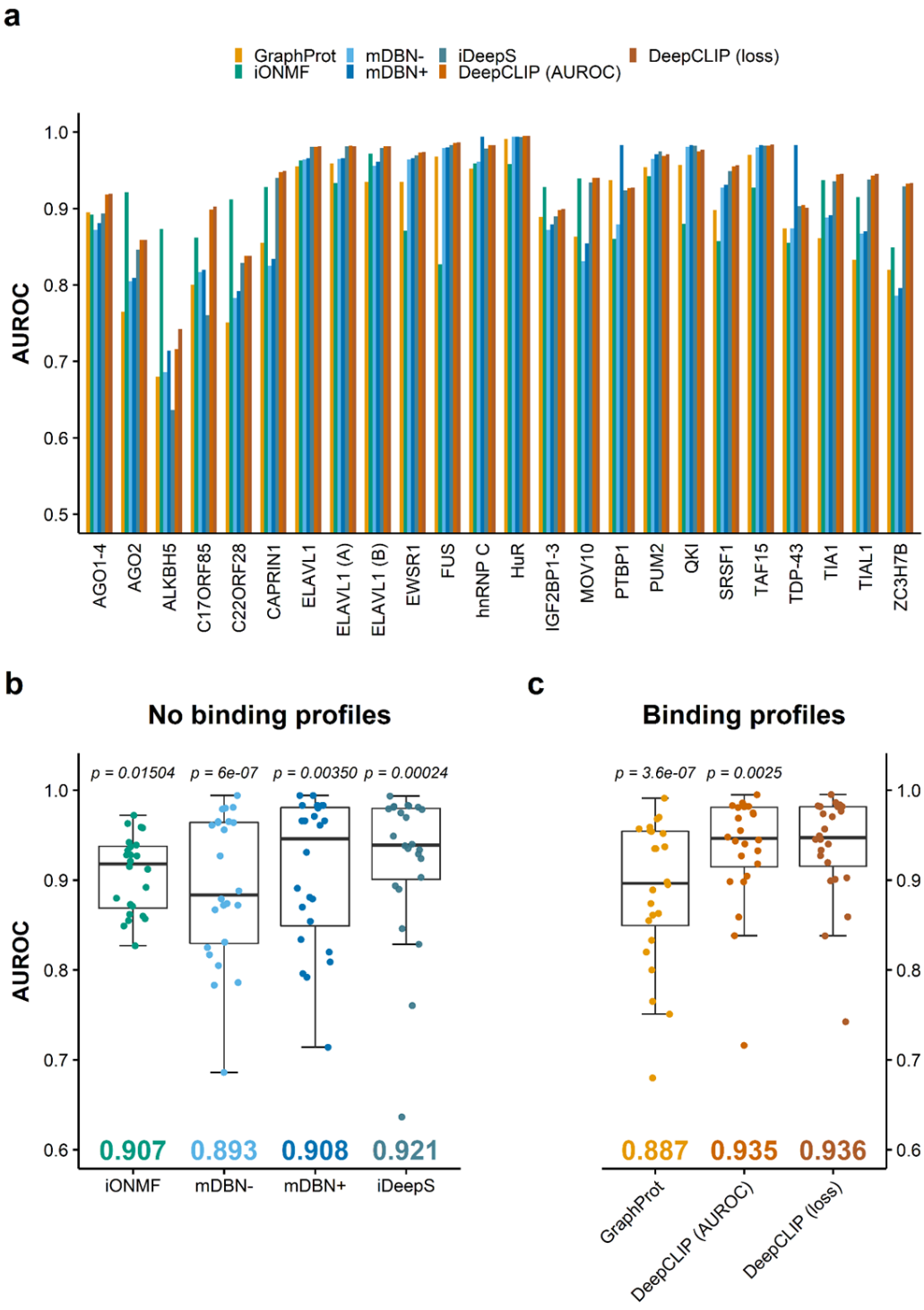


Figure S3 | Comparison of DeepCLIP with loss-selection on a benchmark dataset with existing tools. (a) Same as Figure S2a, but with addition of the DeepCLIP loss-based model performance. (b-c) Same as Figure 2C, but with addition of DeepCLIP loss-based model performance, and p-values indicate significance between the method and DeepCLIP with loss-based model selection.

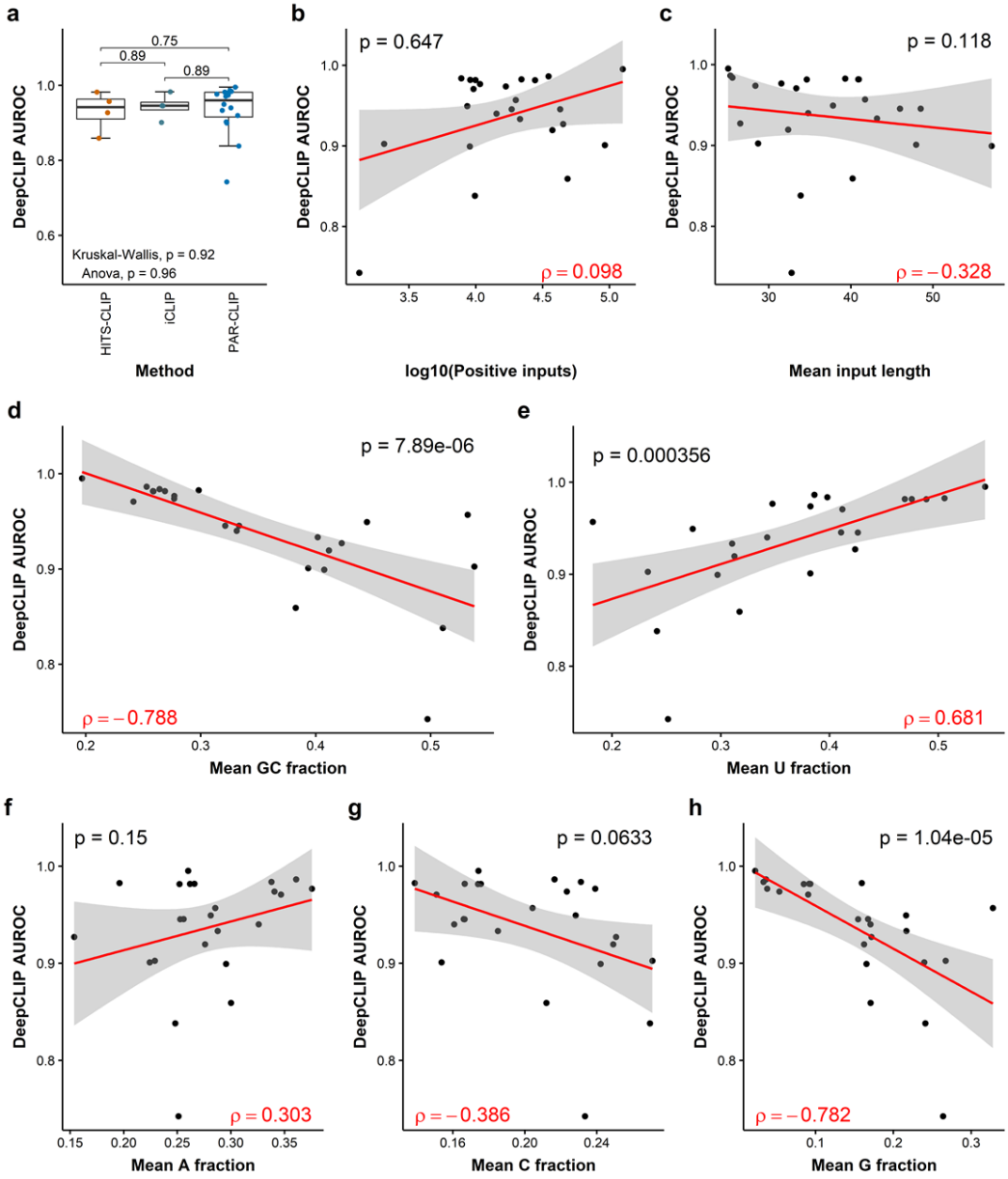


Figure S4 | DeepCLIP AUROC performance by CLIP method and dataset size. (a) AUROC metrics for DeepCLIP grouped into CLIP methods. Significance of pairwise differences was calculated using Wilcoxon rank sum tests. (b) Correlation plot of DeepCLIP AUROC measures and log10 of the number of input binding sites. Correlation was estimated using Spearman's rank correlation rho. (c) Same as (b), but correlation between mean input length and AUROC. (d) Correlation between GC-percentage and AUROC. (e) Correlation between uracil-percentage and AUROC. (f-h), same as (e), but with A-percentage, C.-percentage, and G-percentage respectively.

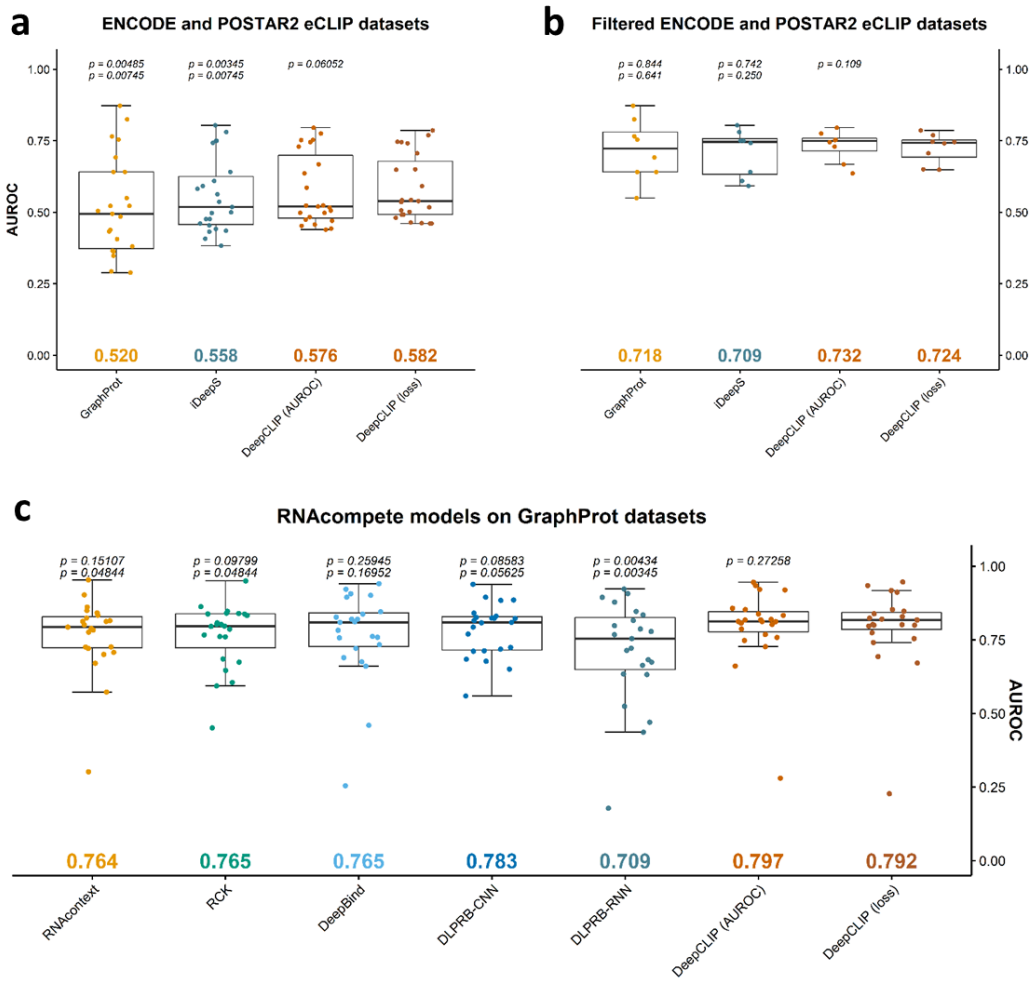


Figure S5 | Benchmark of models on independent datasets. (a) Model performance on eCLIP datasets from POSTAR2 and ENCODE measured by AUROC of GraphProt, iDeepS, DeepCLIP with AUROC based best model selection, and DeepCLIP with loss based best-model selection. The top row p-values are comparison against DeepCLIP (loss), and the bottom row against DeepCLIP (AUROC). The mean of each method is indicated at the bottom of the plot. (b) Same as (a), but only those dataset where at least one model showed AUROC > 0.6. (c) DeepCLIP models trained on the top and bottom 1000 sequences from RNAcompete experiments previously used to compare models on the GraphProt datasets (DLPRB paper). Numbers were taken from the previous study and the DeepCLIP models' performance were measured on the GraphProt datasets. At the bottom the mean AUROC performance of each method is indicated. The top row p-values are comparison (paired Wilcoxon) against DeepCLIP with loss-based model selection, and the bottom row p-values are comparison against DeepCLIP with AUROC-based model selection.

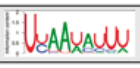
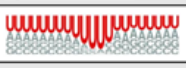
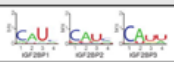
Protein (dataset)	Known binding preference	Add. binding preference source	CLIP data source	CLIP method	Top-2 CNN filters
AGO1-4	Mainly binds coding regions and 3'UTR	-	Hafner et al., 2010	PAR-CLIP	 
AGO2	Mainly binds coding regions and 3'UTR	-	Kishore et al., 2011	HITS-CLIP	 
ALKBH5	Mainly binds coding regions and 3'UTR	-	Baltz et al., 2012	PAR-CLIP	 
C17ORF85	Mainly binds coding regions	-	Baltz et al., 2012	PAR-CLIP	 
C22ORF28	Mainly binds coding regions	-	Baltz et al., 2012	PAR-CLIP	 
CAPRIN1	Mainly binds coding regions and 3'UTR	-	Baltz et al., 2012	PAR-CLIP	 
ELAVL1		Gao et al, 1994	Kishore et al., 2011	HITS-CLIP	 
ELAVL1 (A)		Gao et al, 1994	Kishore et al., 2011	PAR-CLIP	 
ELAVL1 (B)		Gao et al, 1994	Lebedeva et al., 2011	PAR-CLIP	 
EWSR1	Binds AU-rich loop structures	-	Hoell et al., 2011	PAR-CLIP	 
FUS		Munteanu et al, 2018	Hoell et al., 2011	PAR-CLIP	 
hnRNP C		-	König et al., 2010	iCLIP	 
HuR		Gao et al, 1994	Mukherjee et al., 2011	PAR-CLIP	 
IGF2BP1-3		-	Hafner et al., 2010	PAR-CLIP	 
MOV10	Binds AU-rich 3' UTR regions	Gregersen et al., 2014	Sievers et al., 2012	PAR-CLIP	 
PTBP1		Perez et al., 1997	Xue et al., 2009	HITS-CLIP	 
PUM2		-	Hafner et al., 2010	PAR-CLIP	
QKI		-	Hafner et al., 2010	PAR-CLIP	
SRSF1		Feng et al, 2019	Sanford et al., 2009	HITS-CLIP	
TAF15	Binds AU-rich loop structures	-	Hoell et al., 2011	PAR-CLIP	
TDP-43		Colombrita et al, 2012	Tollervey et al., 2011	iCLIP	
TIA1	Binds U-rich regions near 5'ss	Meyer et al, 2018	Wang et al., 2010	iCLIP	
TIAL1	Binds U-rich regions near 5'ss	-	Wang et al., 2010	iCLIP	
ZC3H7B	Mainly binds coding regions, 3'UTR and intronic regions	-	Baltz et al., 2012	PAR-CLIP	

Figure S6 | Pseudo-PFMs captured by DeepCLIP. The column Protein contains the names of the analyzed protein/dataset, which have binding preferences as described in the column Known binding preference. References to literature can be seen in column 3 and 4. In the outermost right column, the top-2 by information content score Pseudo-PFM, motifs derived from DeepCLIPs convolutional filters are shown. H: A, C or U; N: A, C, G or U.

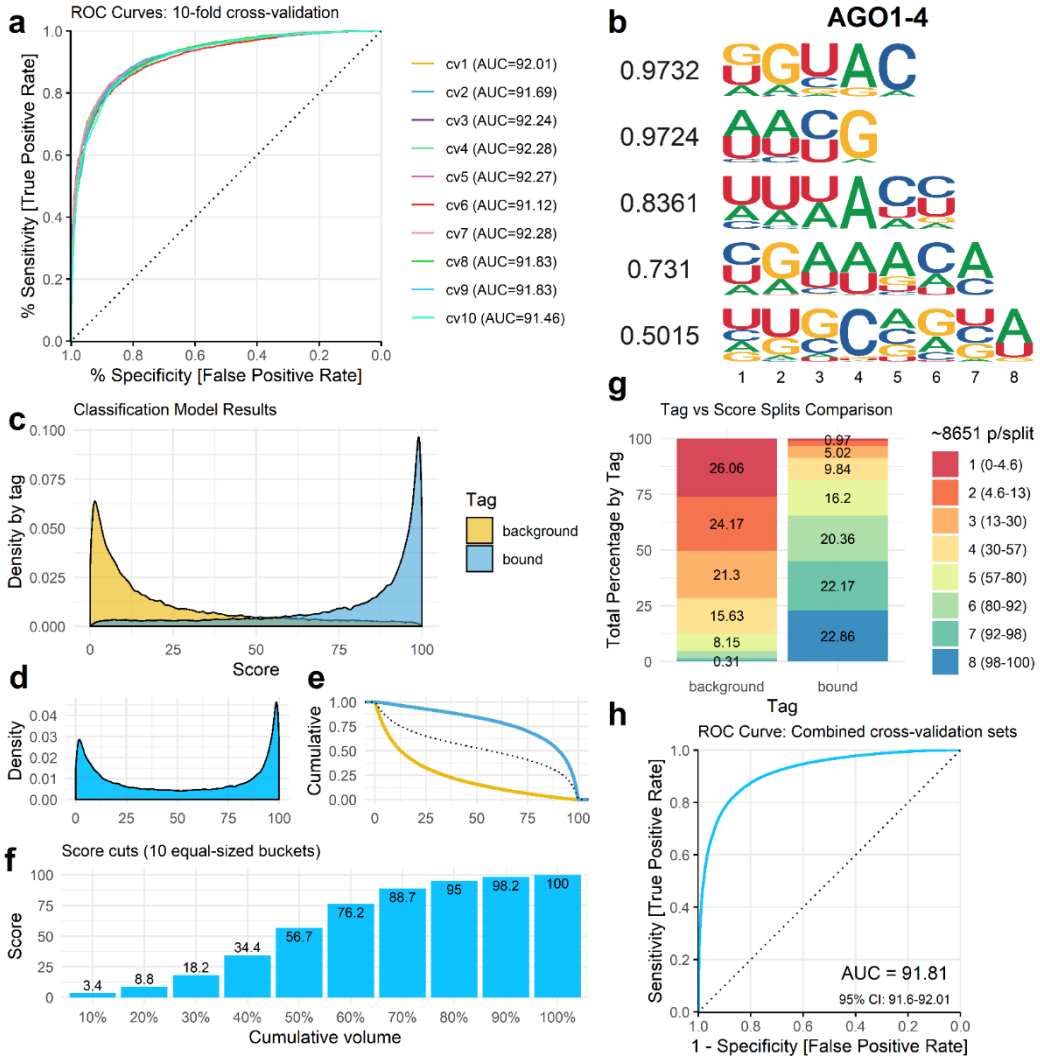


Figure S7 | DeepCLIP model characteristics for AGO1-4. (a) Area under curve analysis of DeepCLIP models trained on AGO1-4 PAR-CLIP data from Hafner et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

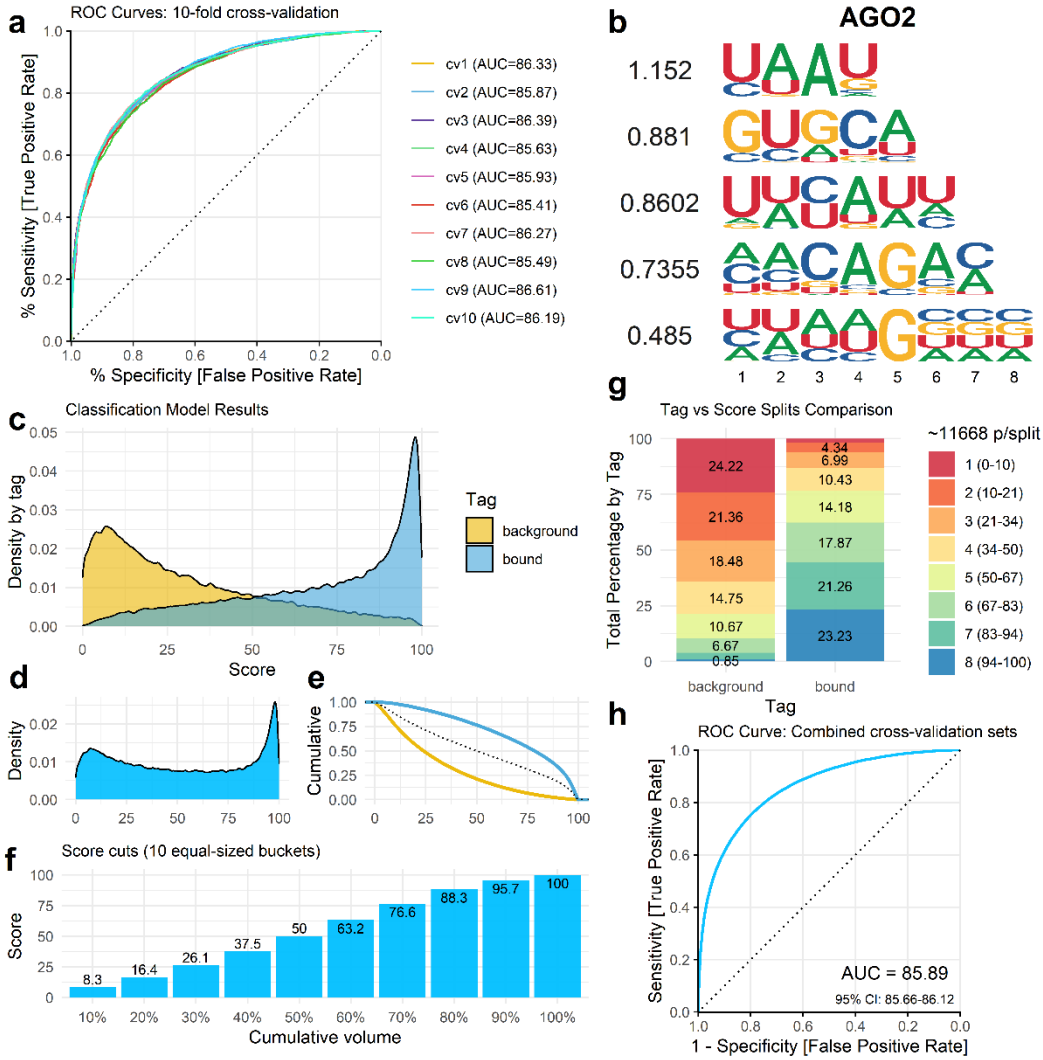


Figure S8 | DeepCLIP model characteristics for AGO2. (a) Area under curve analysis of DeepCLIP models trained on AGO2 HITS-CLIP data from Kishore et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

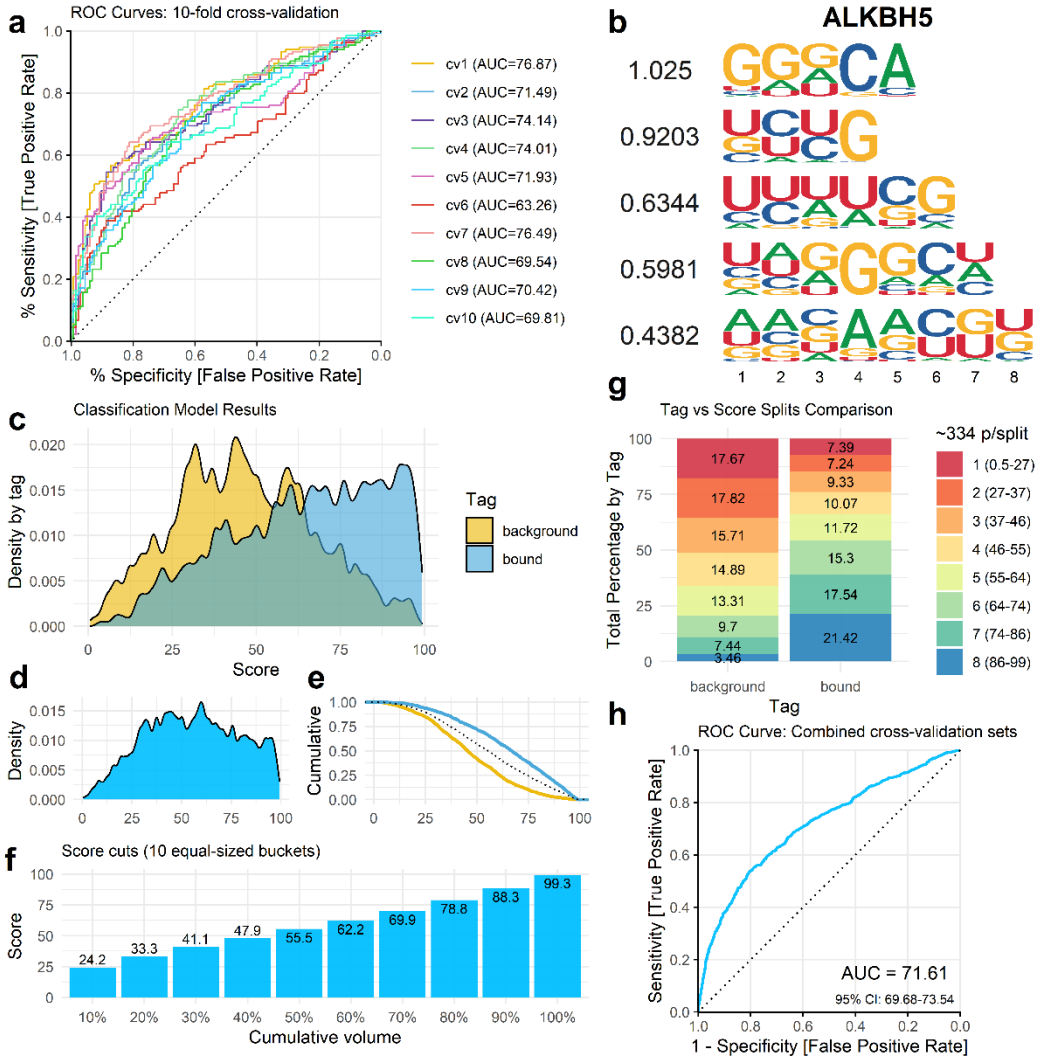


Figure S9 | DeepCLIP model characteristics for ALKBH5. (a) Area under curve analysis of DeepCLIP models trained on ALKBH5 PAR-CLIP data from Baltz et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

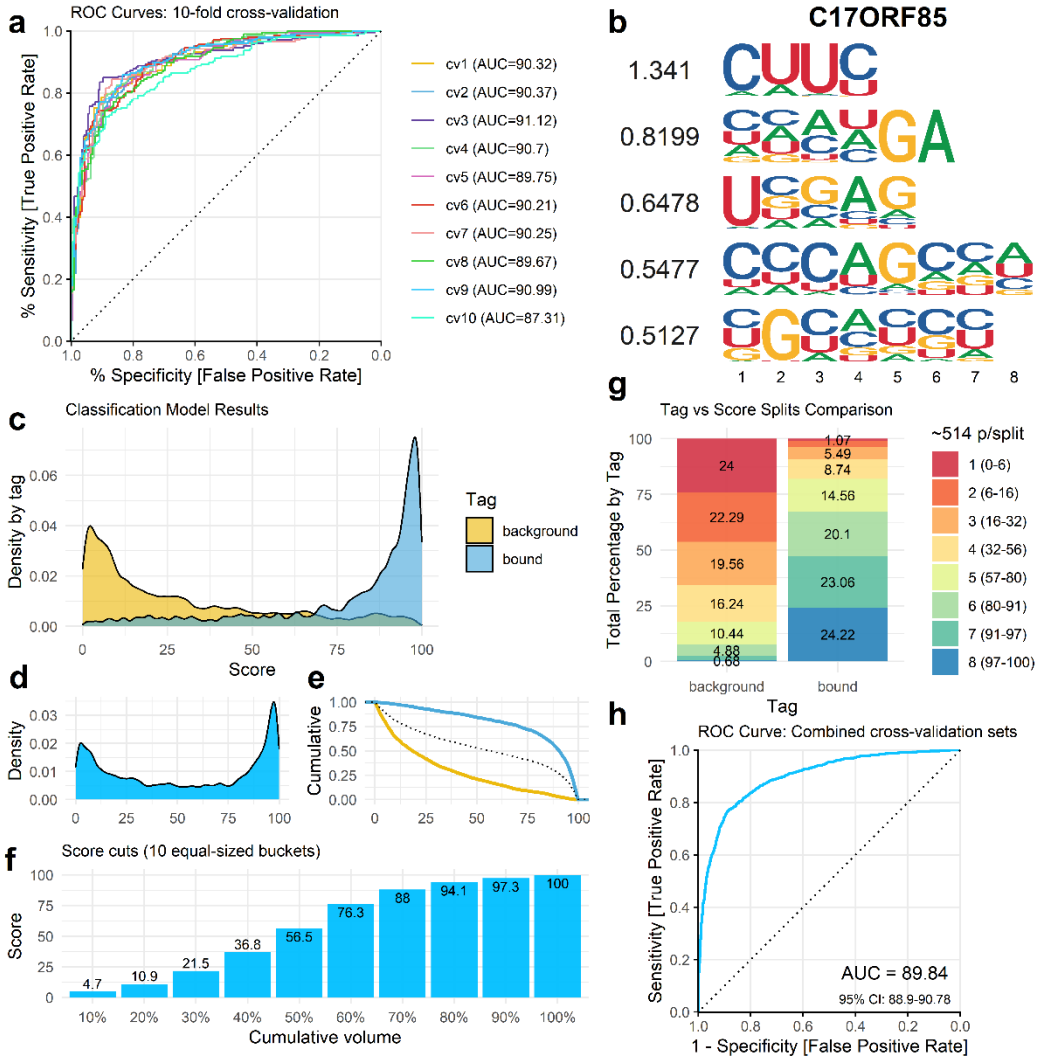


Figure S10 | DeepCLIP model characteristics for C17ORF85. (a) Area under curve analysis of DeepCLIP models trained on C17ORF85 PAR-CLIP data from Baltz et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

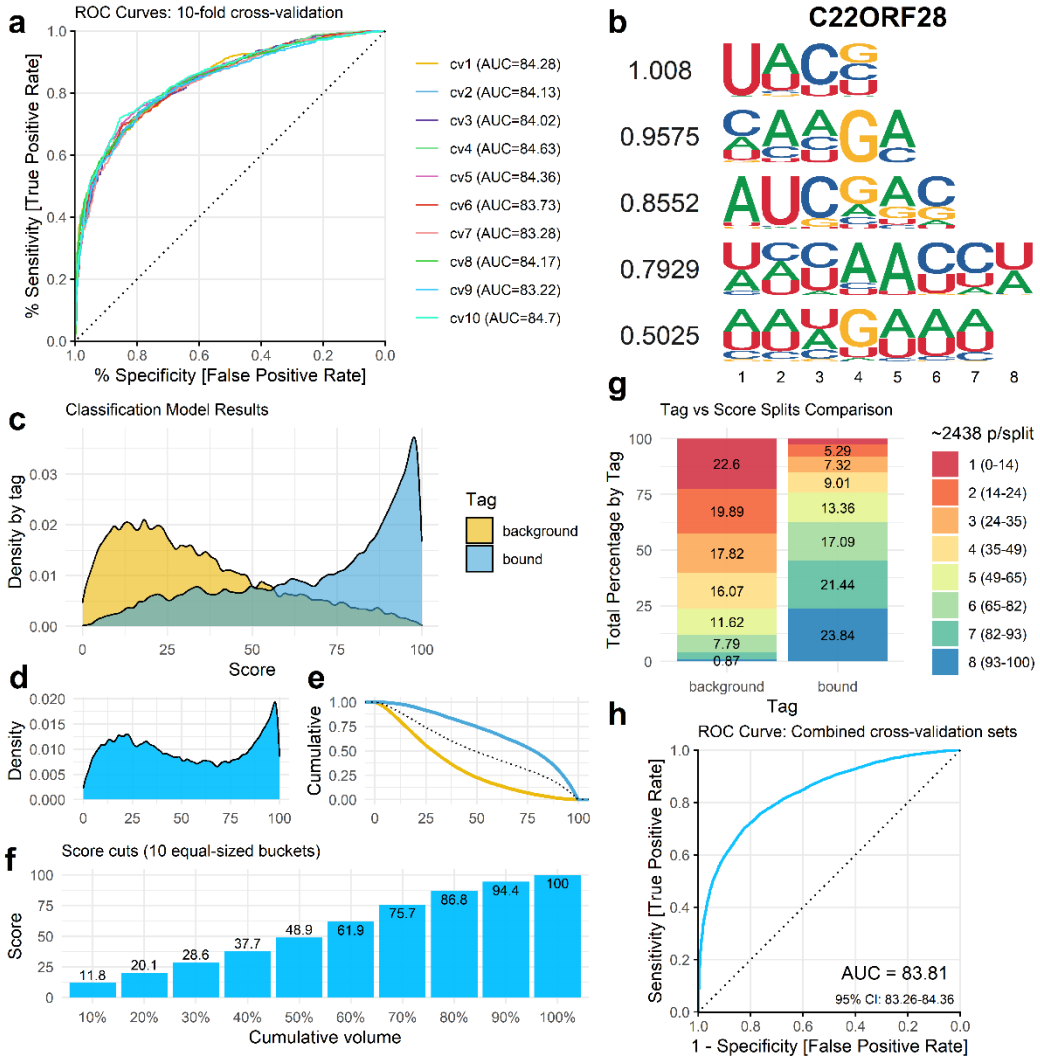


Figure S11 | DeepCLIP model characteristics for C22ORF28. (a) Area under curve analysis of DeepCLIP models trained on C22ORF28 PAR-CLIP data from Baltz et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

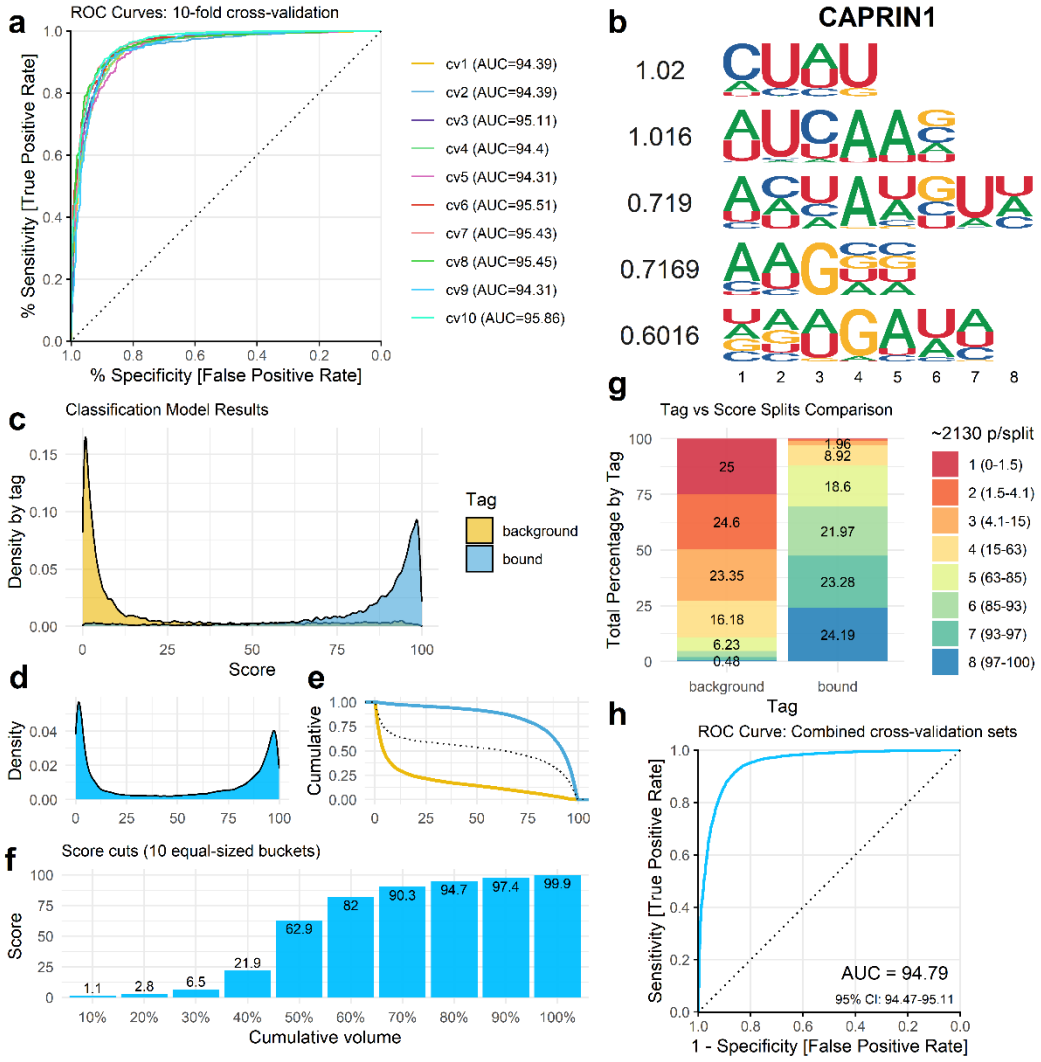


Figure S12 | DeepCLIP model characteristics for CAPRIN1. (a) Area under curve analysis of DeepCLIP models trained on CAPRIN1 PAR-CLIP data from Baltz et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

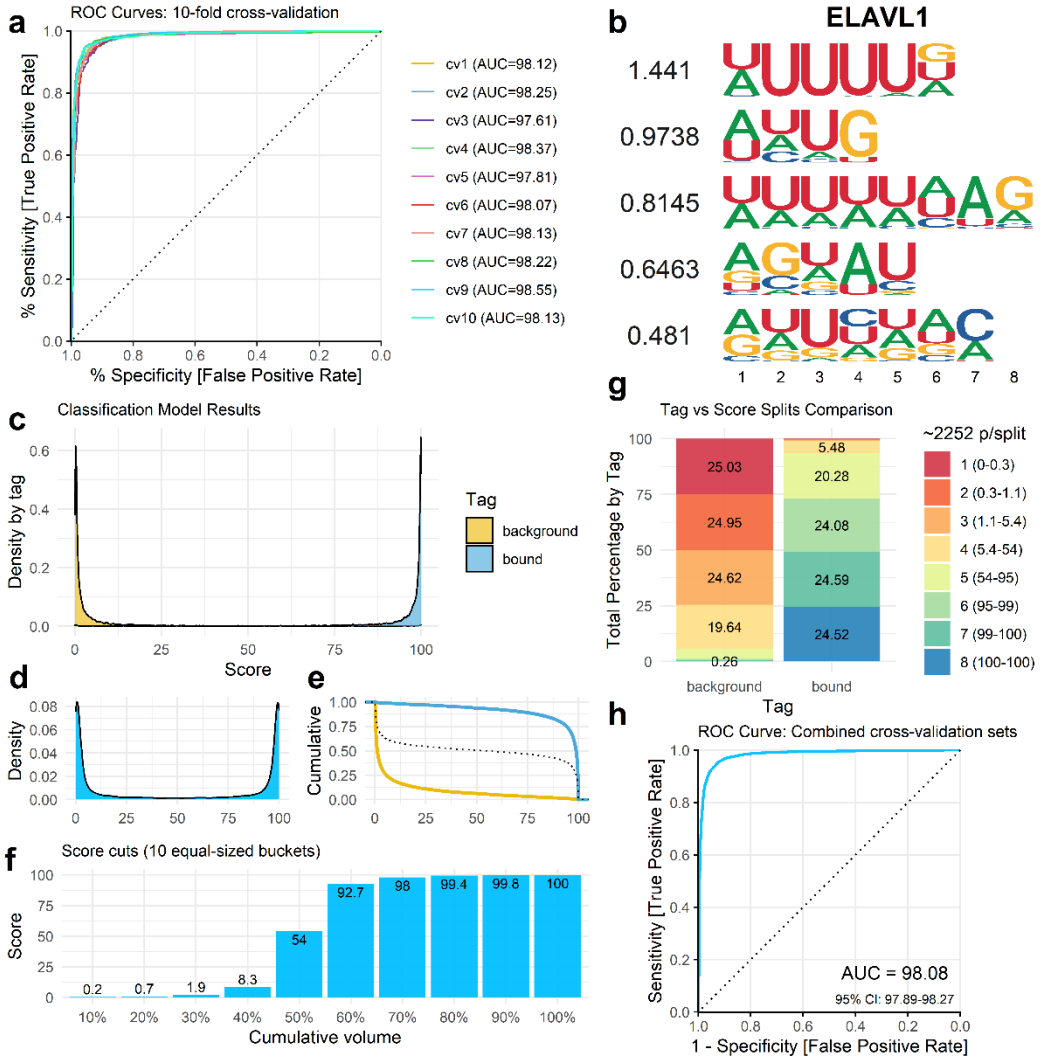


Figure S13 | DeepCLIP model characteristics for ELAVL1. (a) Area under curve analysis of DeepCLIP models trained on ELAVL1 HITS-CLIP data from Kishore et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

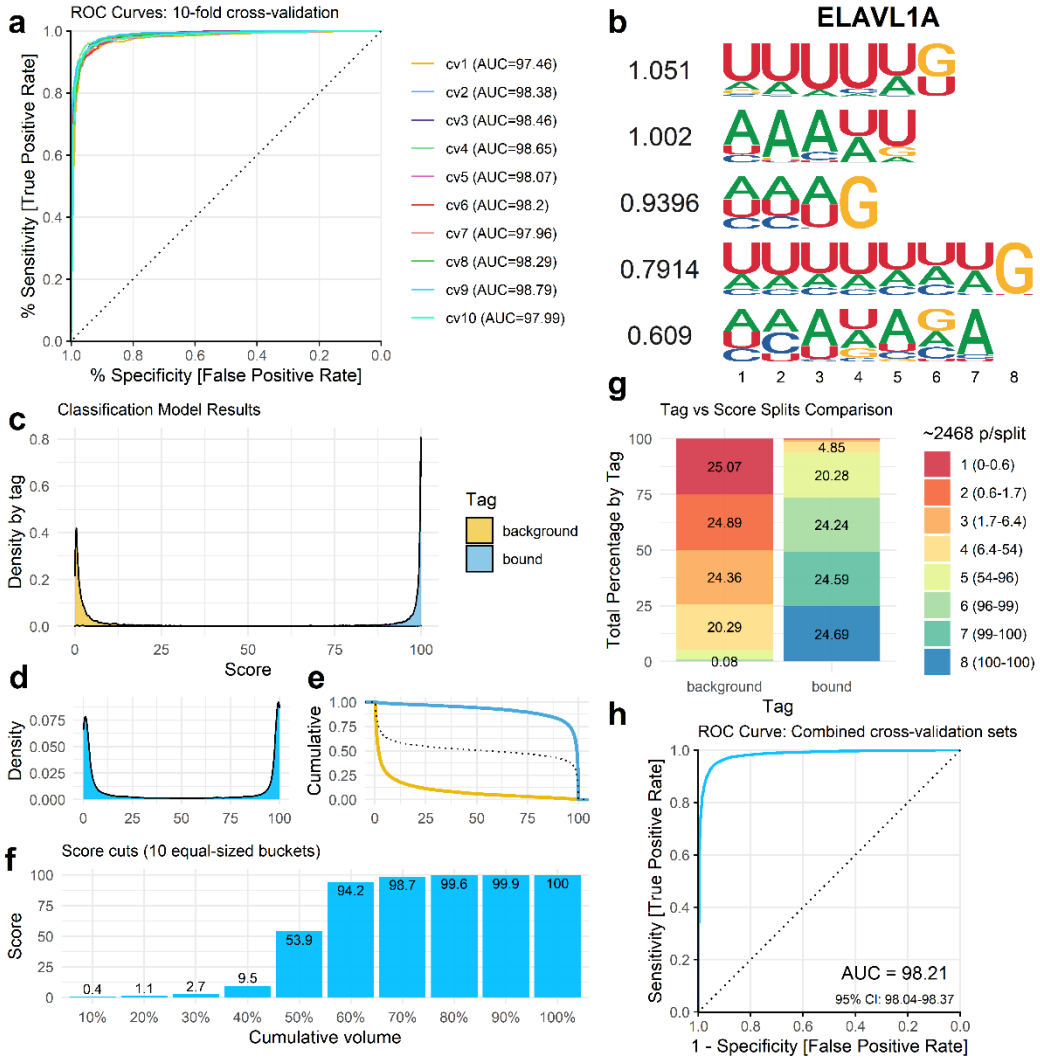


Figure S14 | DeepCLIP model characteristics for ELAVL1A. (a) Area under curve analysis of DeepCLIP models trained on ELAVL1A PAR-CLIP data from Kishore et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

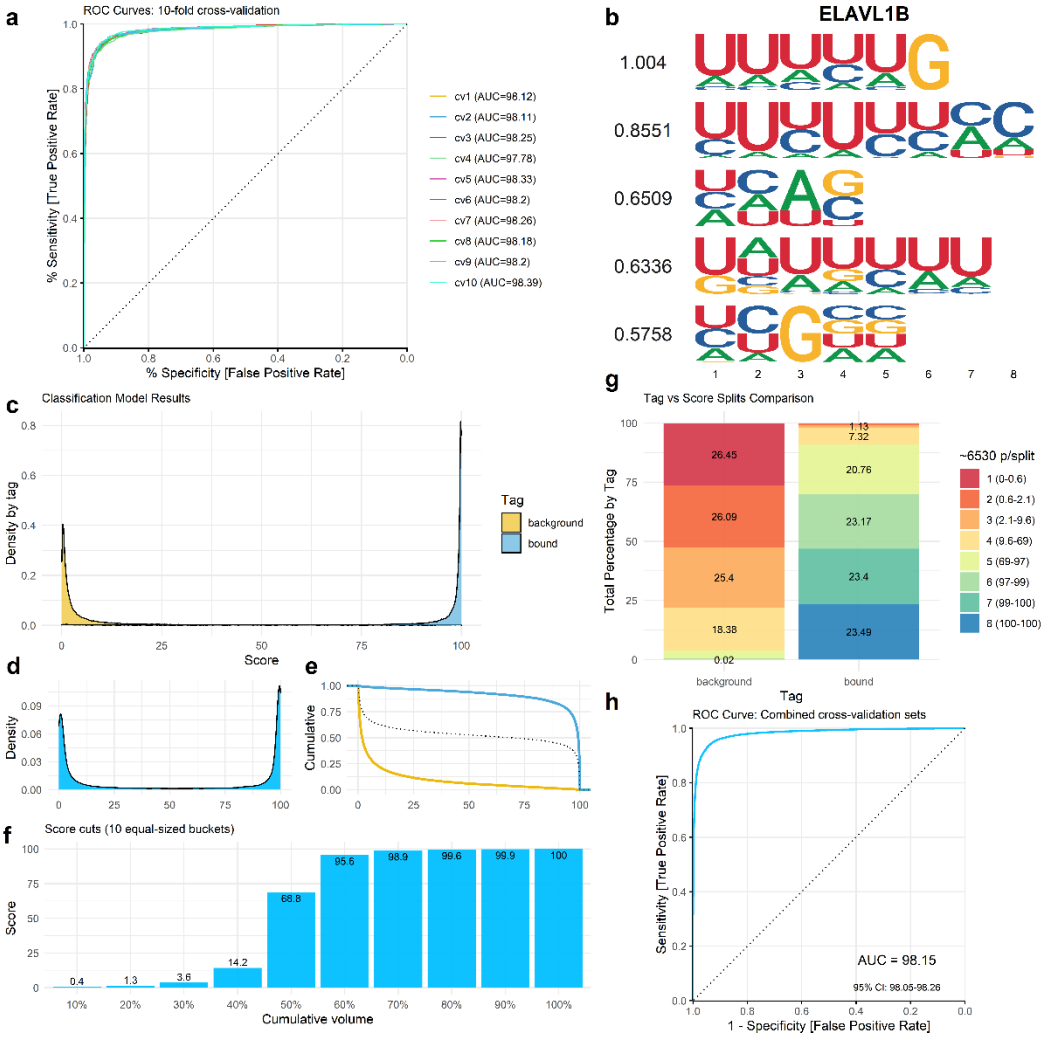


Figure S15 | DeepCLIP model characteristics for ELAVL1B. (a) Area under curve analysis of DeepCLIP models trained on ELAVL1B PAR-CLIP data from Lebedeva et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

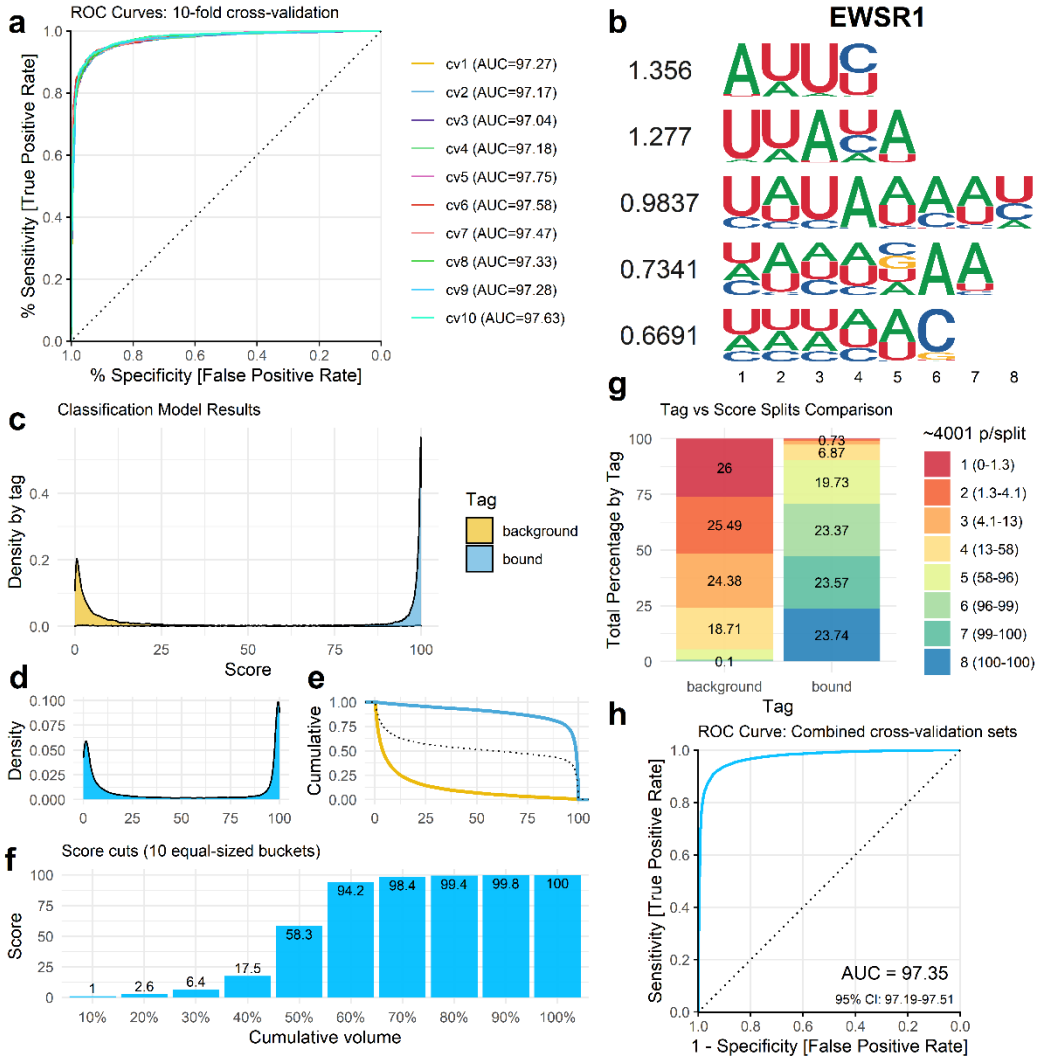


Figure S16 | DeepCLIP model characteristics for EWSR1. (a) Area under curve analysis of DeepCLIP models trained on EWSR1 PAR-CLIP data from Hoell et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

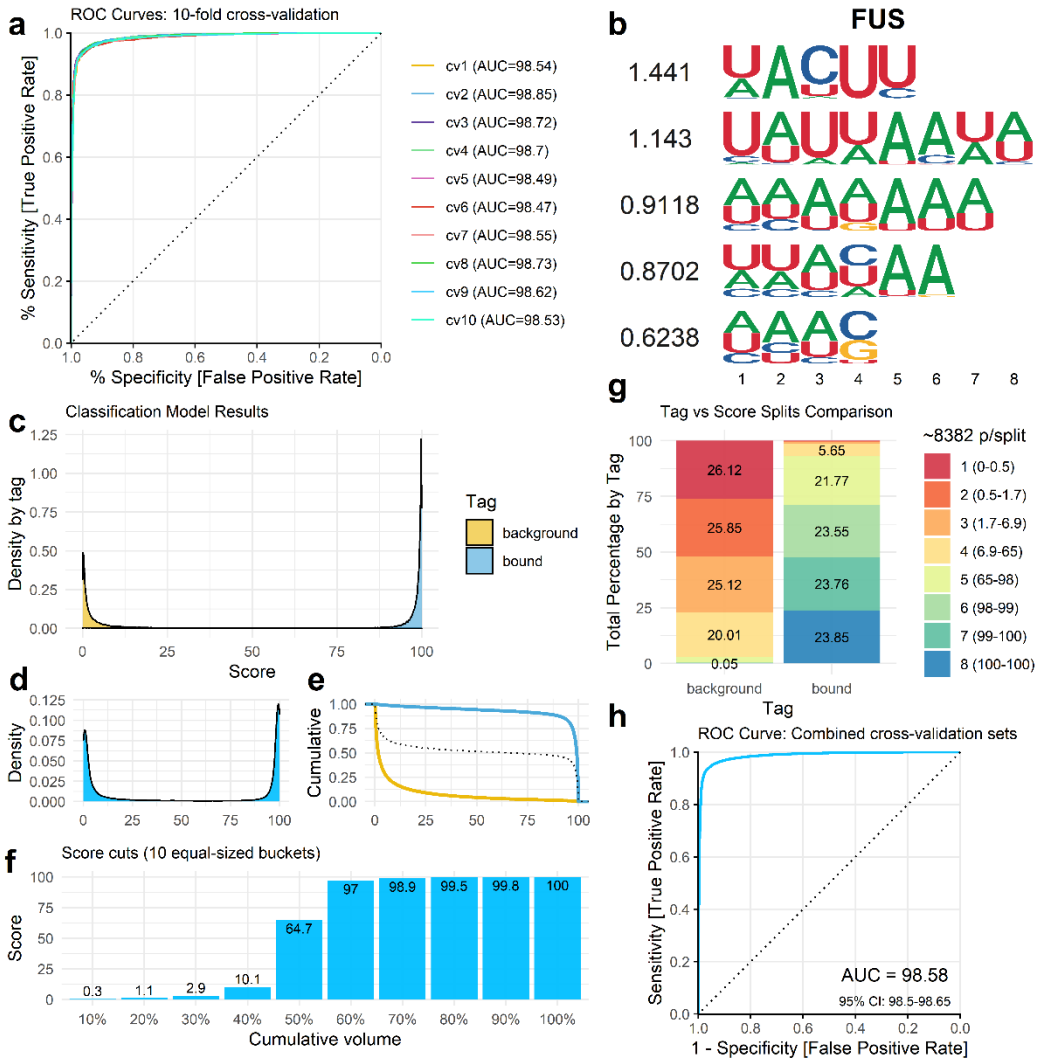


Figure S17 | DeepCLIP model characteristics for FUS. (a) Area under curve analysis of DeepCLIP models trained on FUS PAR-CLIP data from Hoell et al., using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

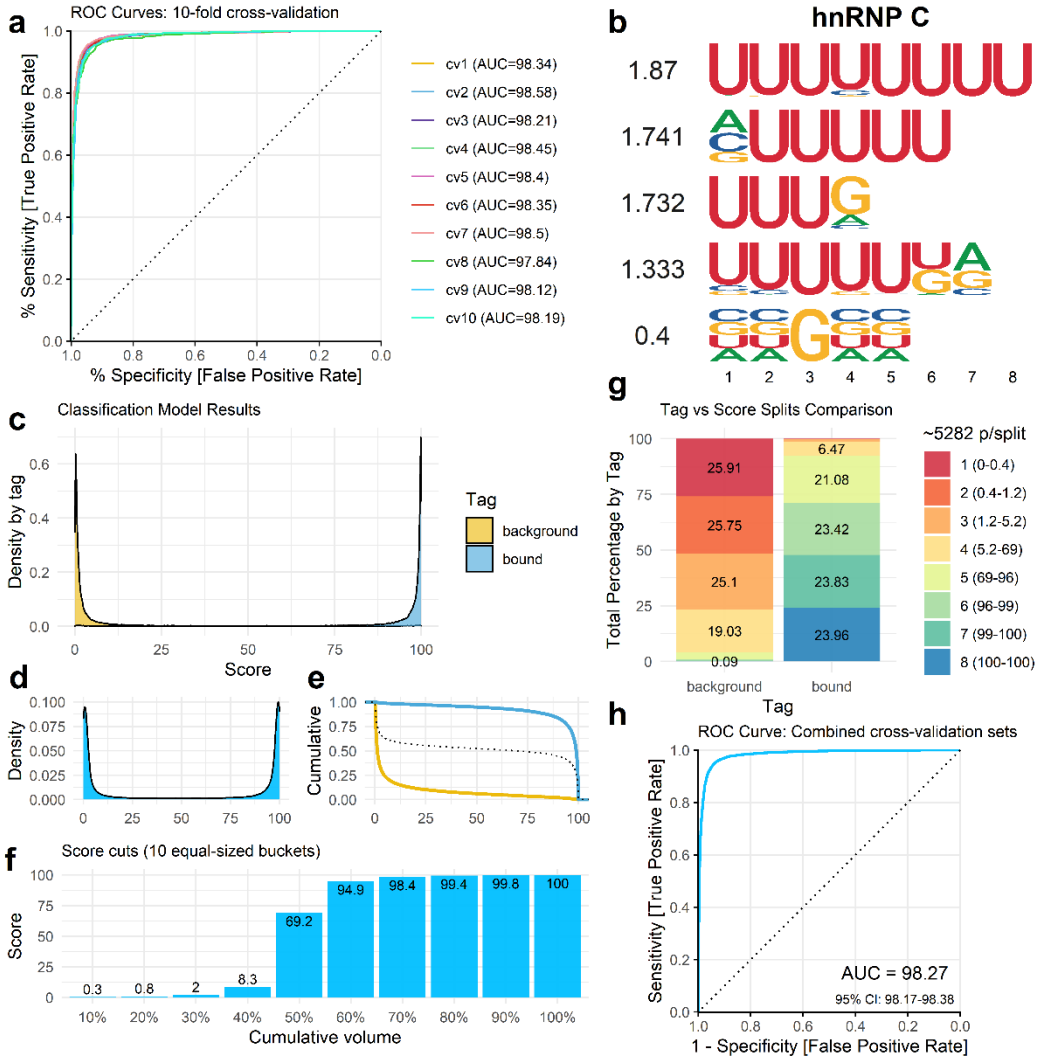


Figure S18 | DeepCLIP model characteristics for hnRNP C. (a) Area under curve analysis of DeepCLIP models trained on hnRNP C iCLIP data from König et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

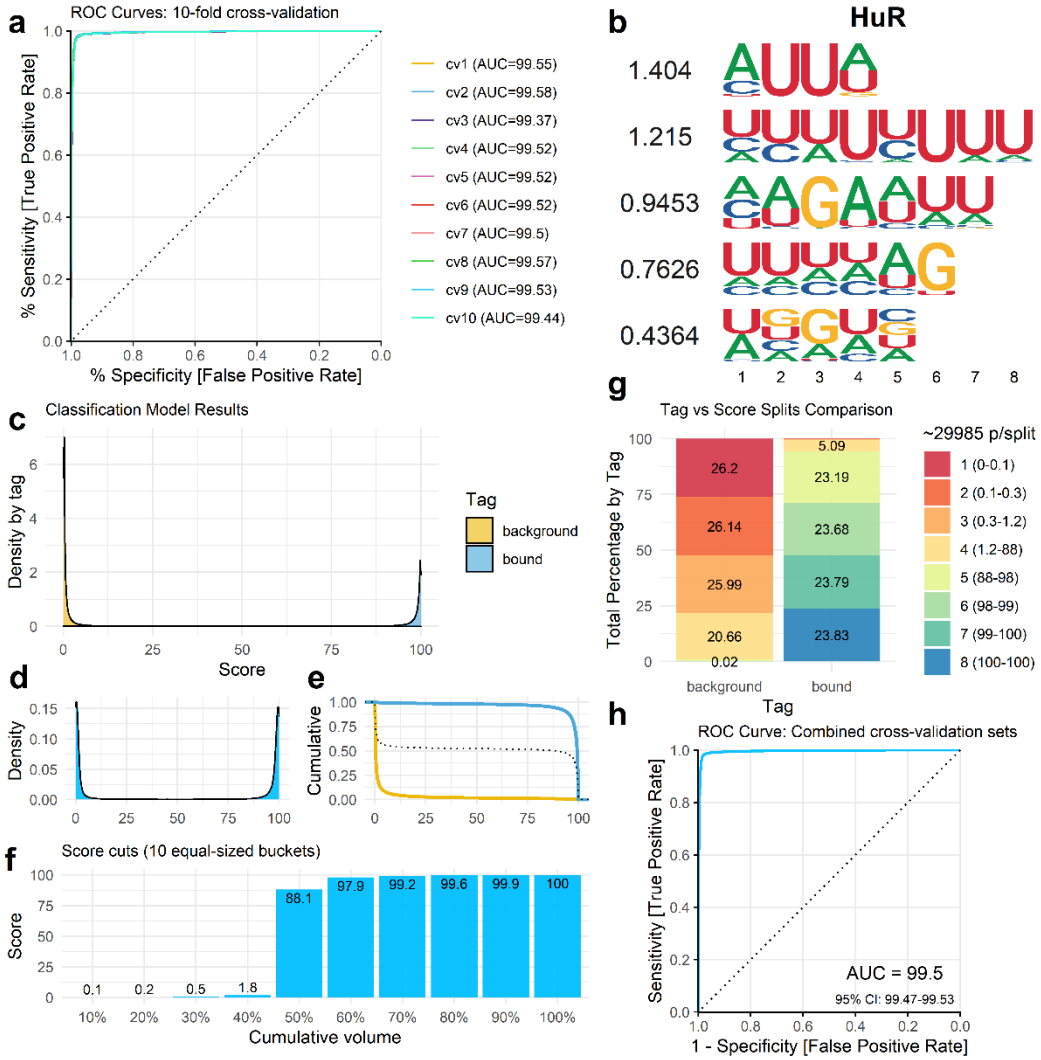


Figure S19 | DeepCLIP model characteristics for HuR. (a) Area under curve analysis of DeepCLIP models trained on HuR PAR-CLIP data from Mukherjee et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

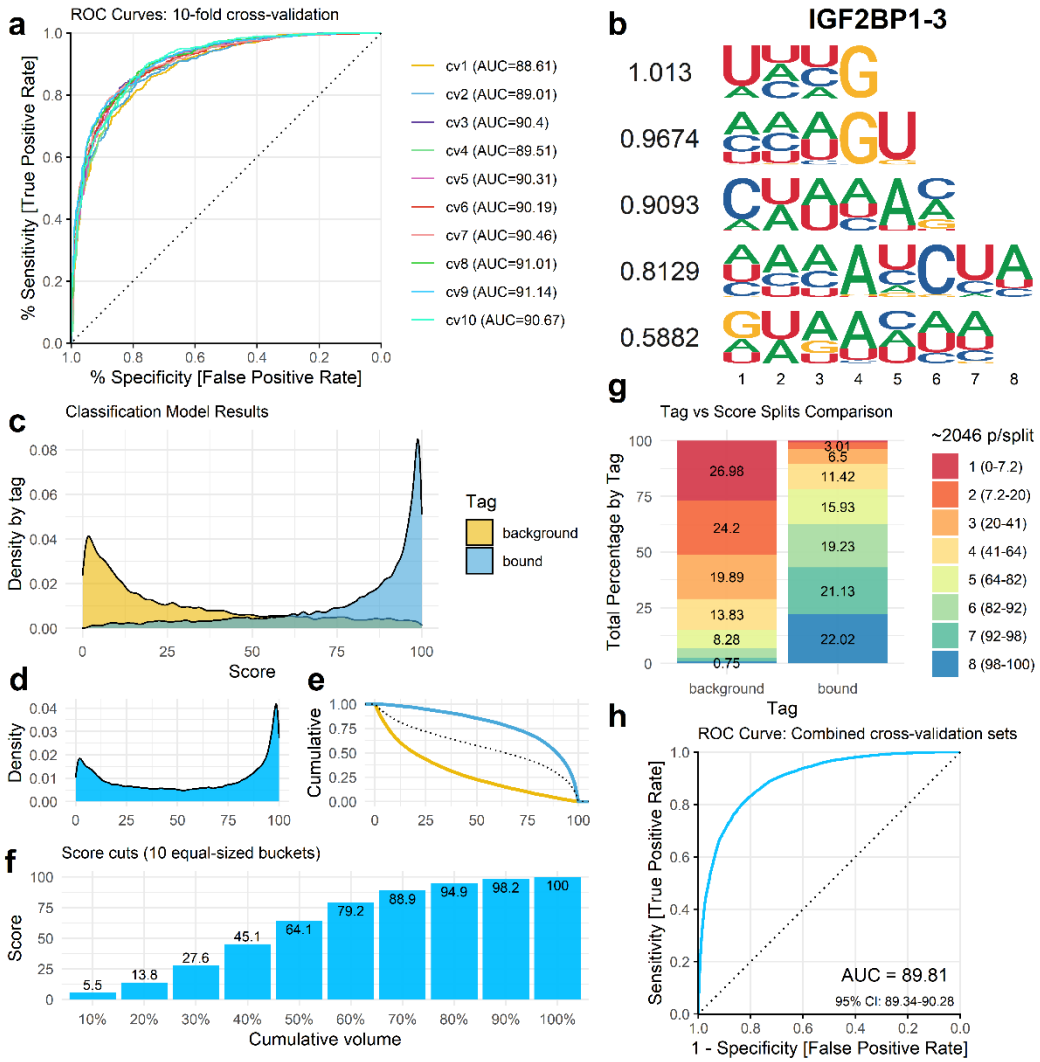


Figure S20 | DeepCLIP model characteristics for IGF2BP1-3. (a) Area under curve analysis of DeepCLIP models trained on IGF2BP1-3 PAR-CLIP data from Hafner et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

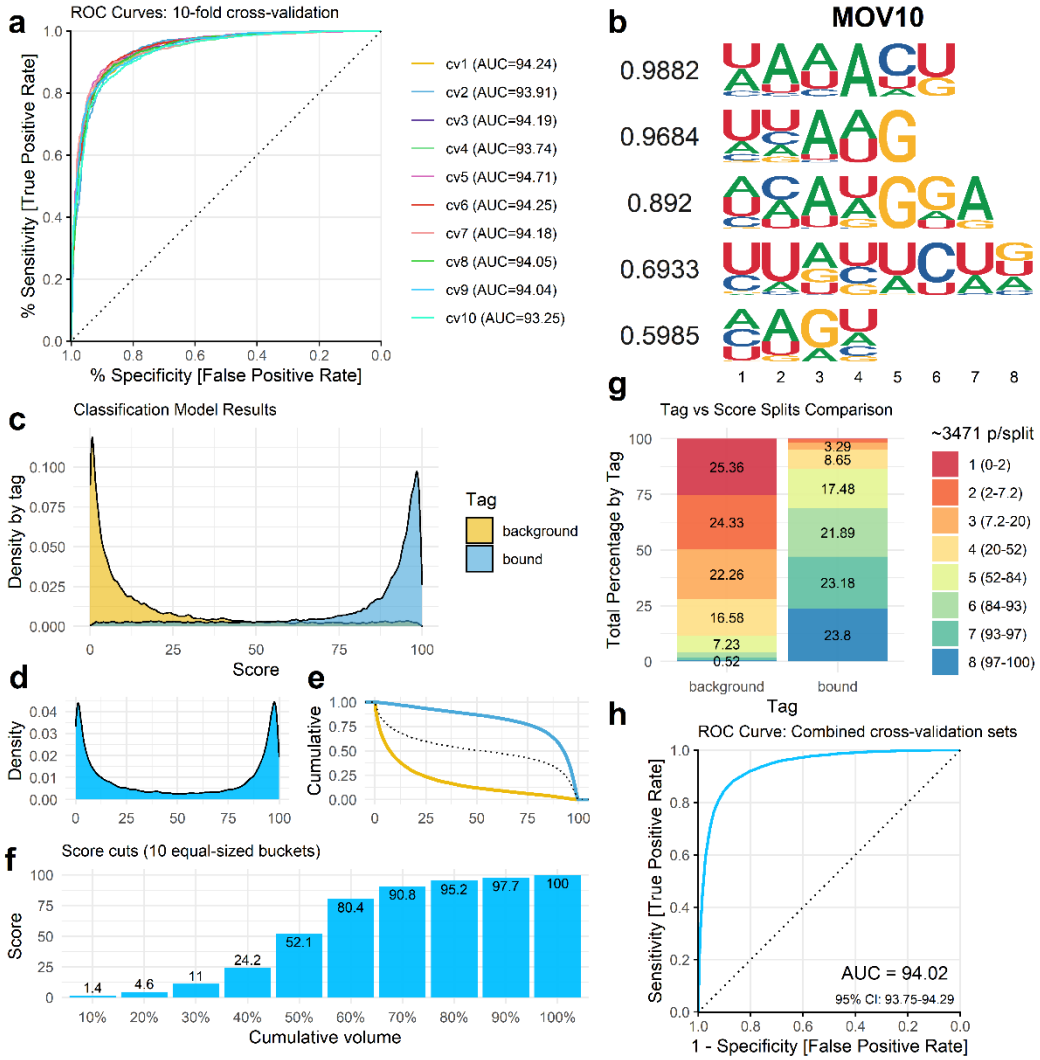


Figure S21 | DeepCLIP model characteristics for MOV10. (a) Area under curve analysis of DeepCLIP models trained on MOV10 PAR-CLIP data from Sievers et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

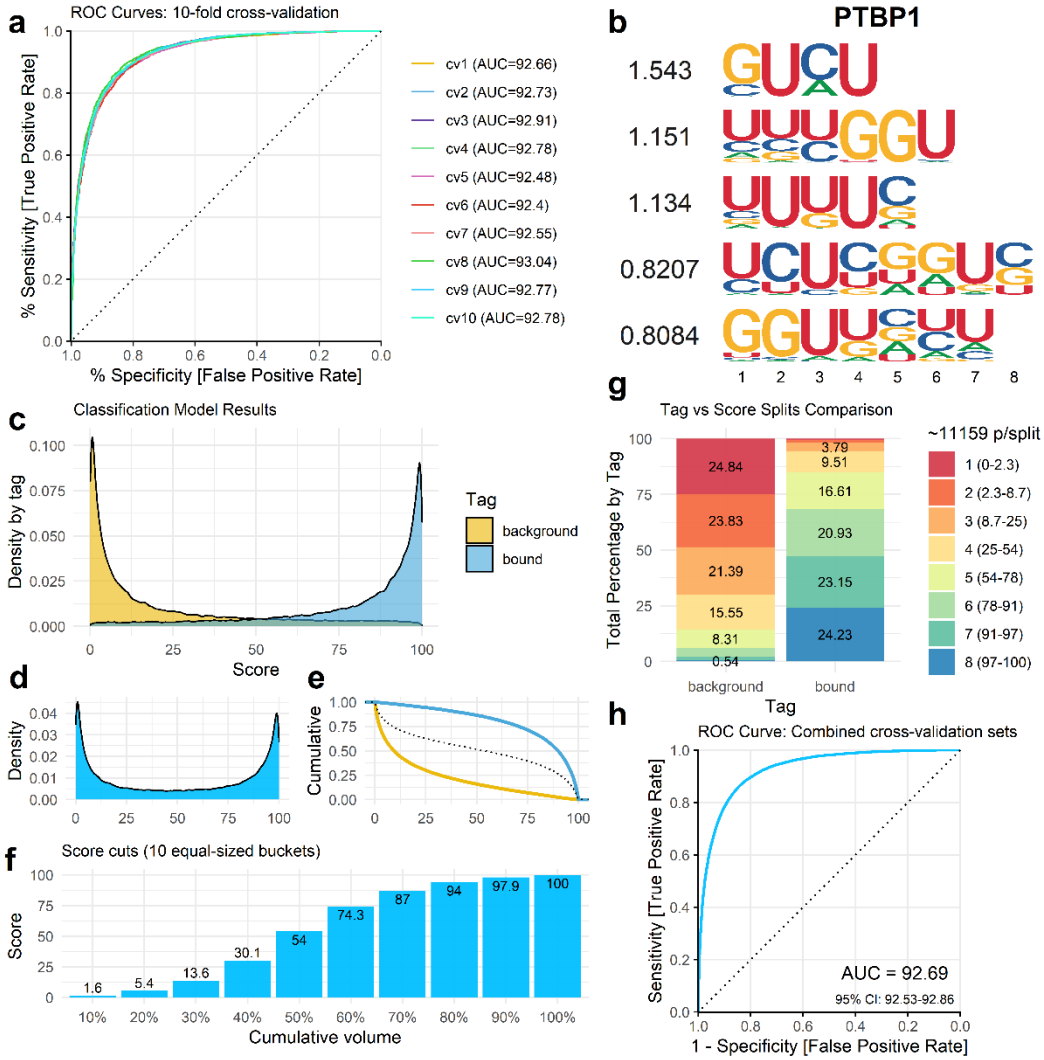


Figure S22 | DeepCLIP model characteristics for PTB1. (a) Area under curve analysis of DeepCLIP models trained on PTB1 HITS-CLIP data from Xue et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

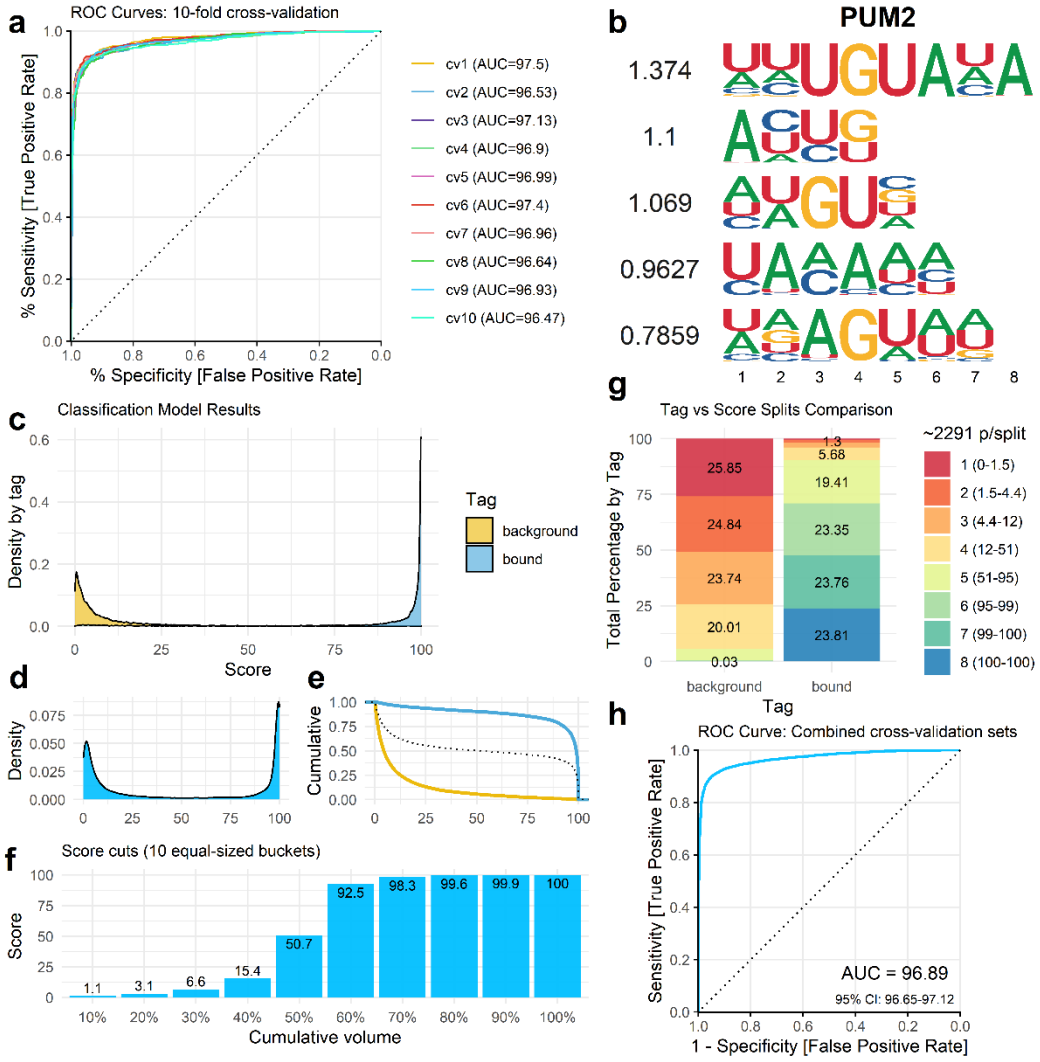


Figure S23 | DeepCLIP model characteristics for PUM2. (a) Area under curve analysis of DeepCLIP models trained on PUM2 PAR-CLIP data from Hafner et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

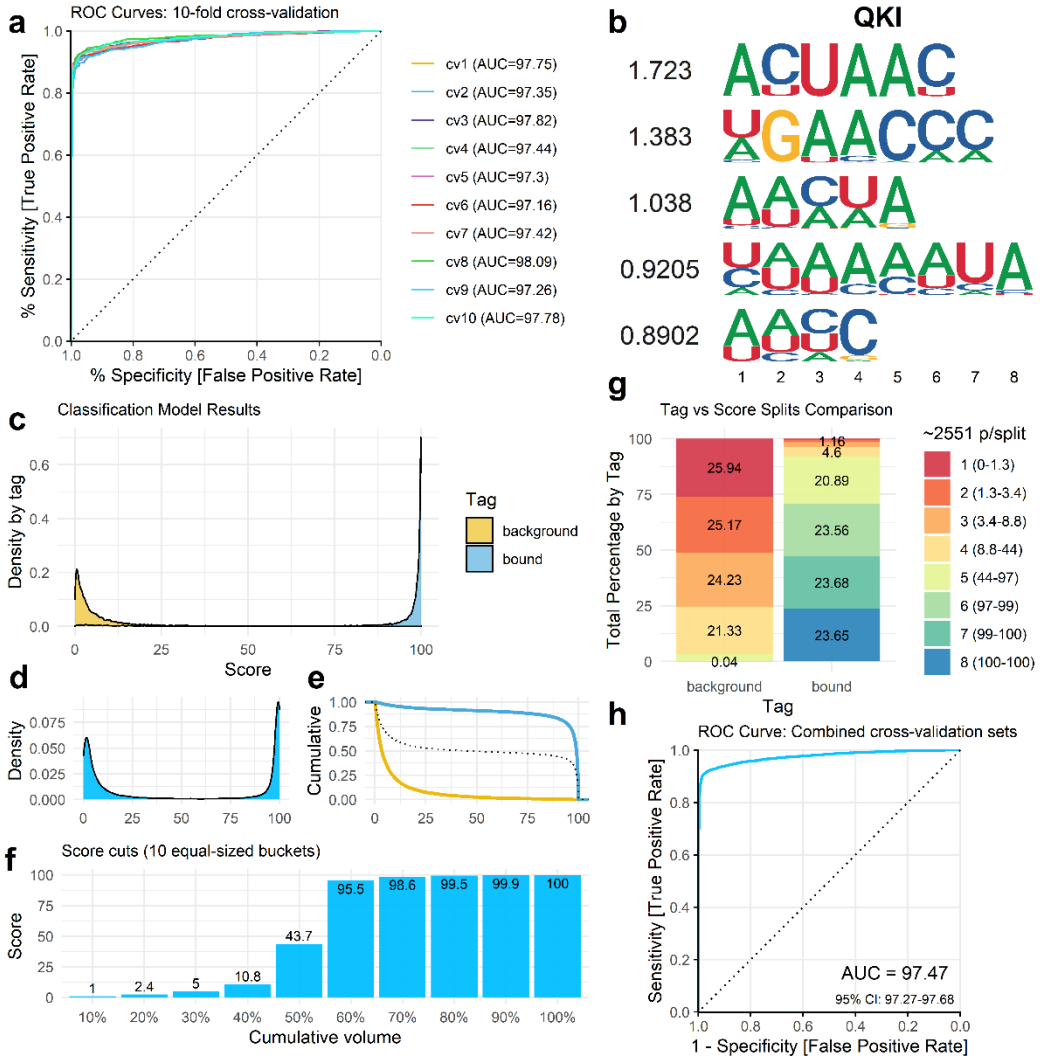


Figure S24 | DeepCLIP model characteristics for QKI. (a) Area under curve analysis of DeepCLIP models trained on QKI PAR-CLIP data from Hafner et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

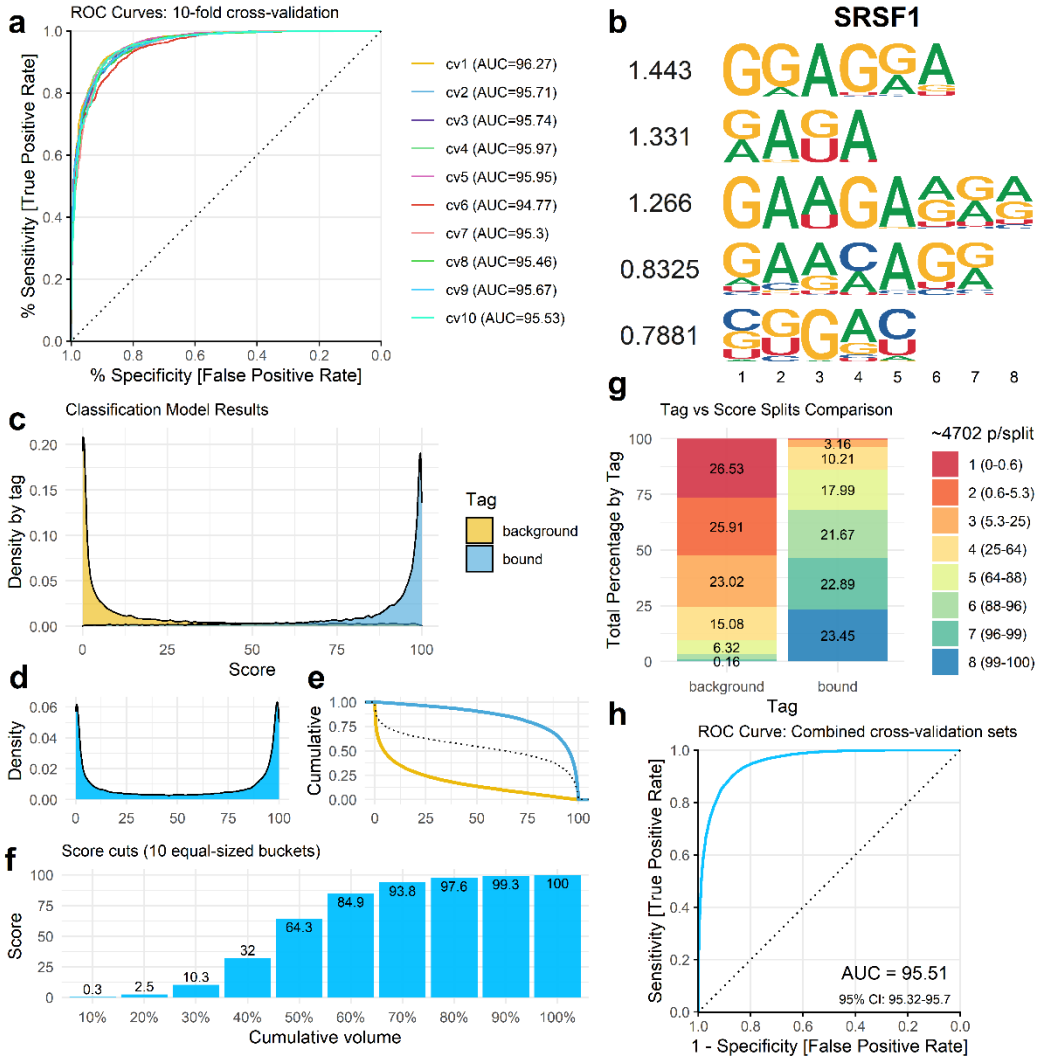


Figure S25 | DeepCLIP model characteristics for SRSF1. (a) Area under curve analysis of DeepCLIP models trained on SRSF1 HITS-CLIP data from Sanford et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

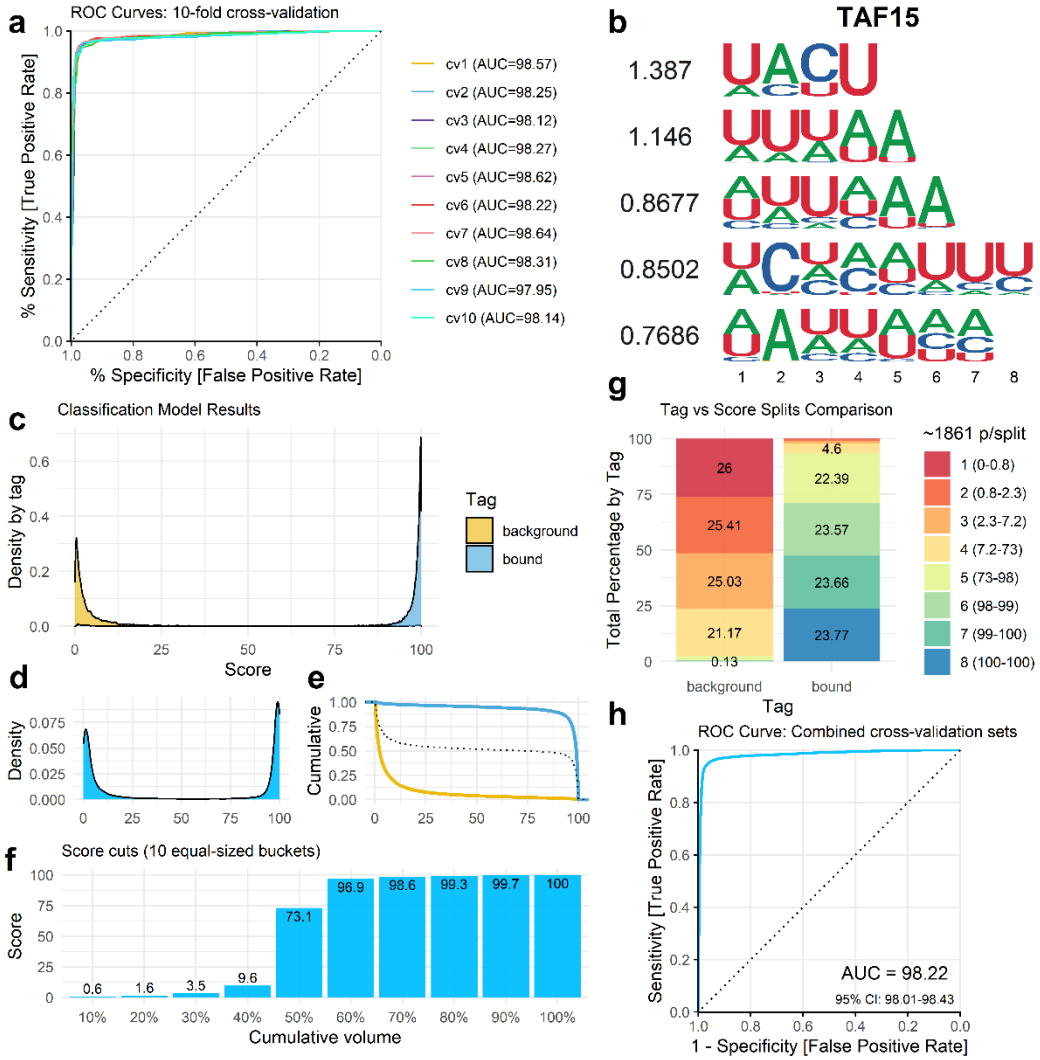


Figure S26 | DeepCLIP model characteristics for TAF15. (a) Area under curve analysis of DeepCLIP models trained on TAF15 PAR-CLIP data from Hoell et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

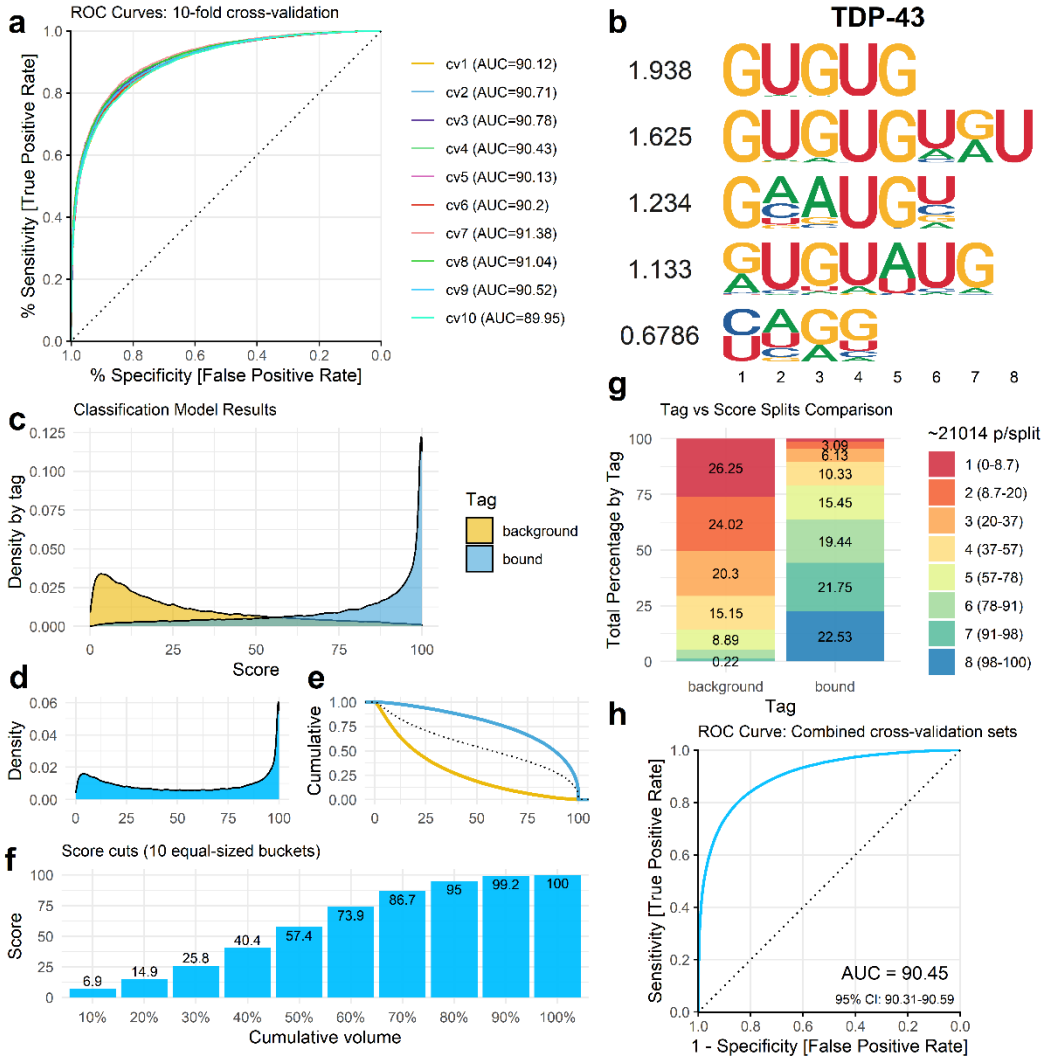


Figure S27 | DeepCLIP model characteristics for TDP-43. (a) Area under curve analysis of DeepCLIP models trained on TDP-43 iCLIP data from Tollervey et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

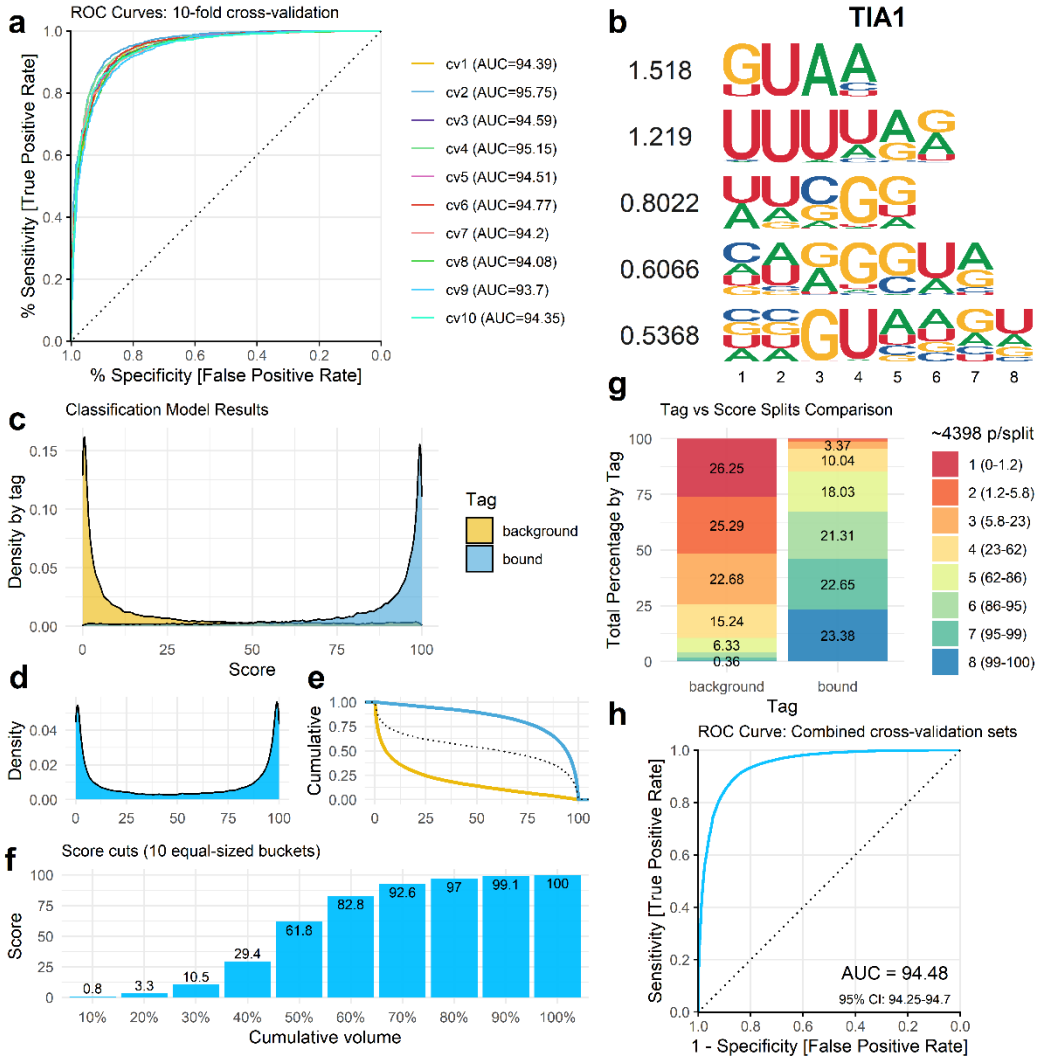


Figure S28 | DeepCLIP model characteristics for TIA1. (a) Area under curve analysis of DeepCLIP models trained on TIA1 iCLIP data from Wang et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

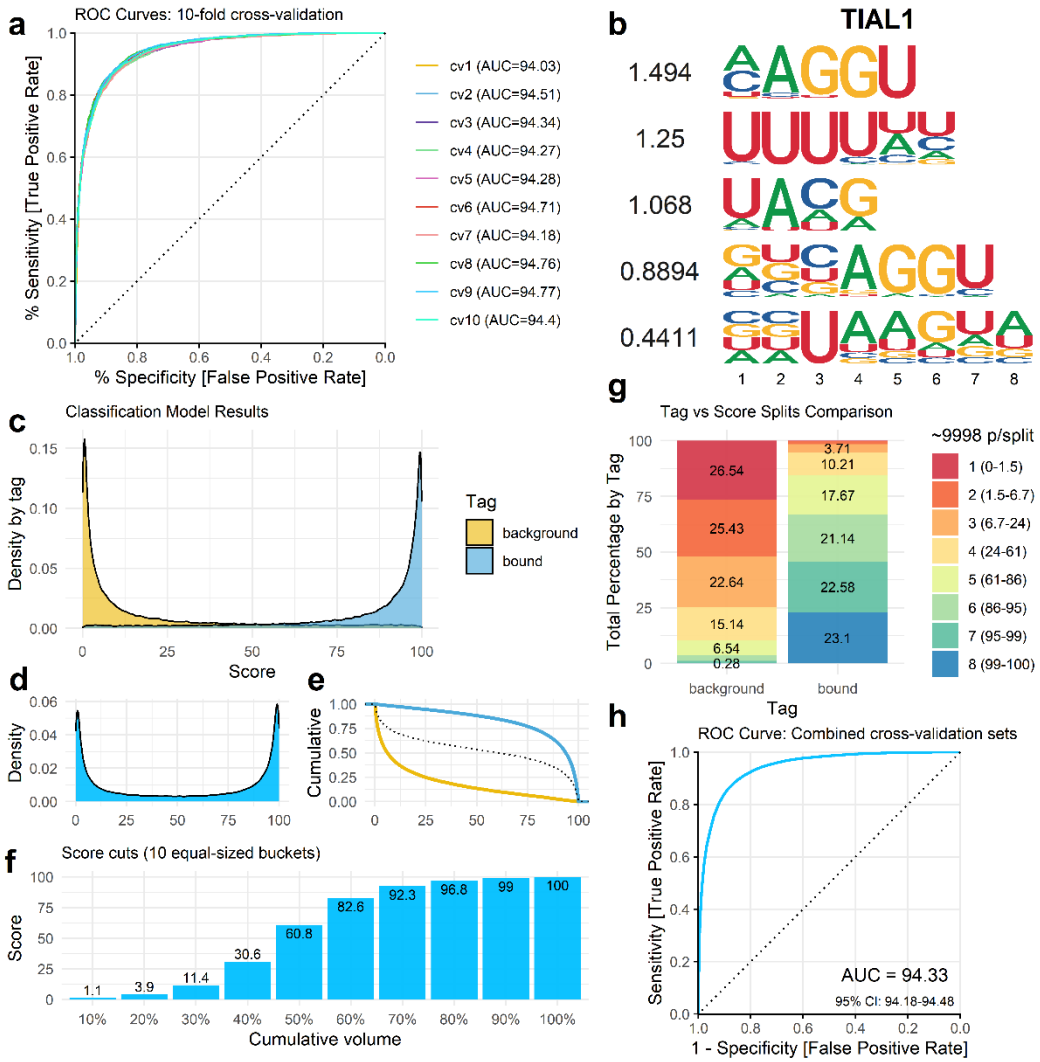


Figure S29 | DeepCLIP model characteristics for TIAL1. (a) Area under curve analysis of DeepCLIP models trained on TIAL1 PAR-CLIP data from Wang et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

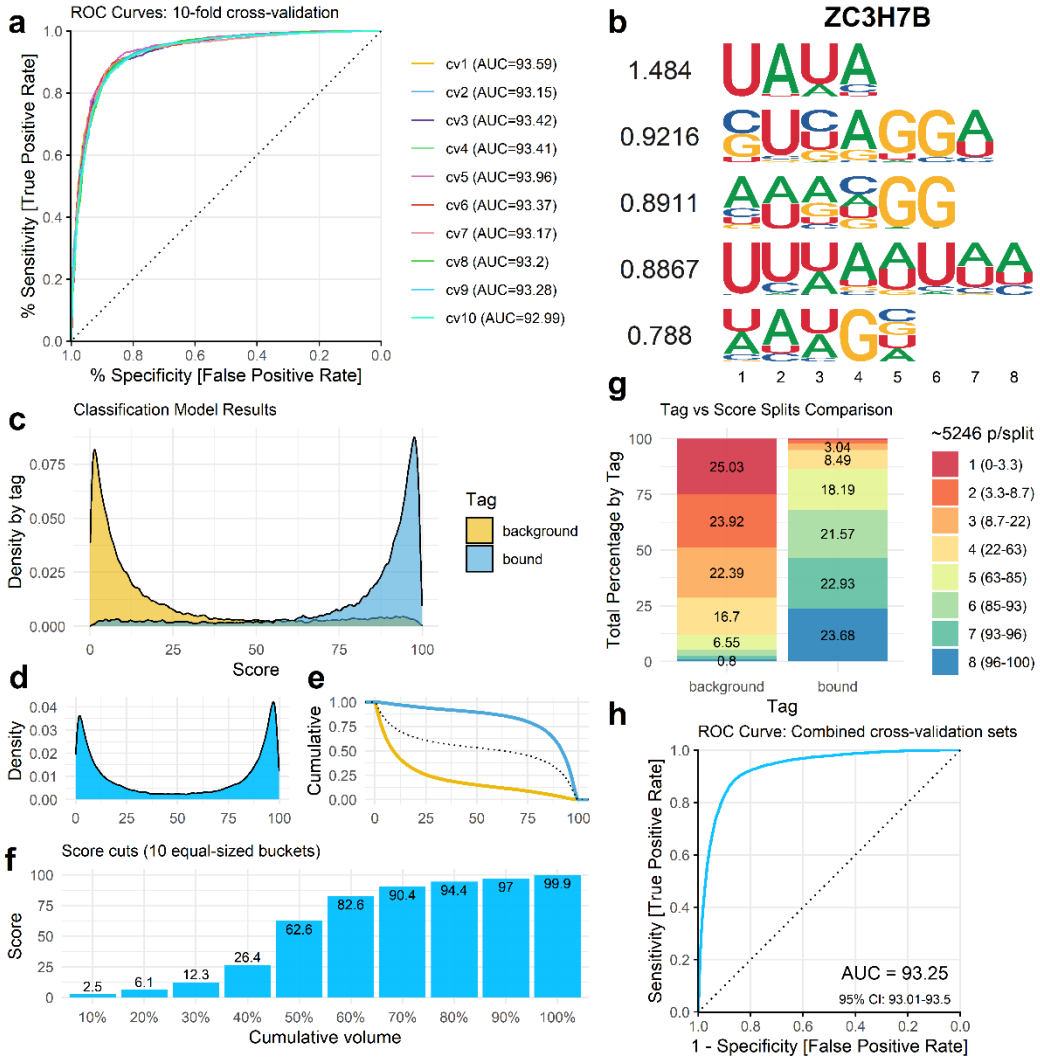


Figure S30 | DeepCLIP model characteristics for ZC3H7B. (a) Area under curve analysis of DeepCLIP models trained on ZC3H7B PAR-CLIP data from Baltz et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

Figure S31 | Alternative CNN filter extraction. CNN filters produced from the sequences among the total training set data, both positives and negatives, scoring above 0.5.

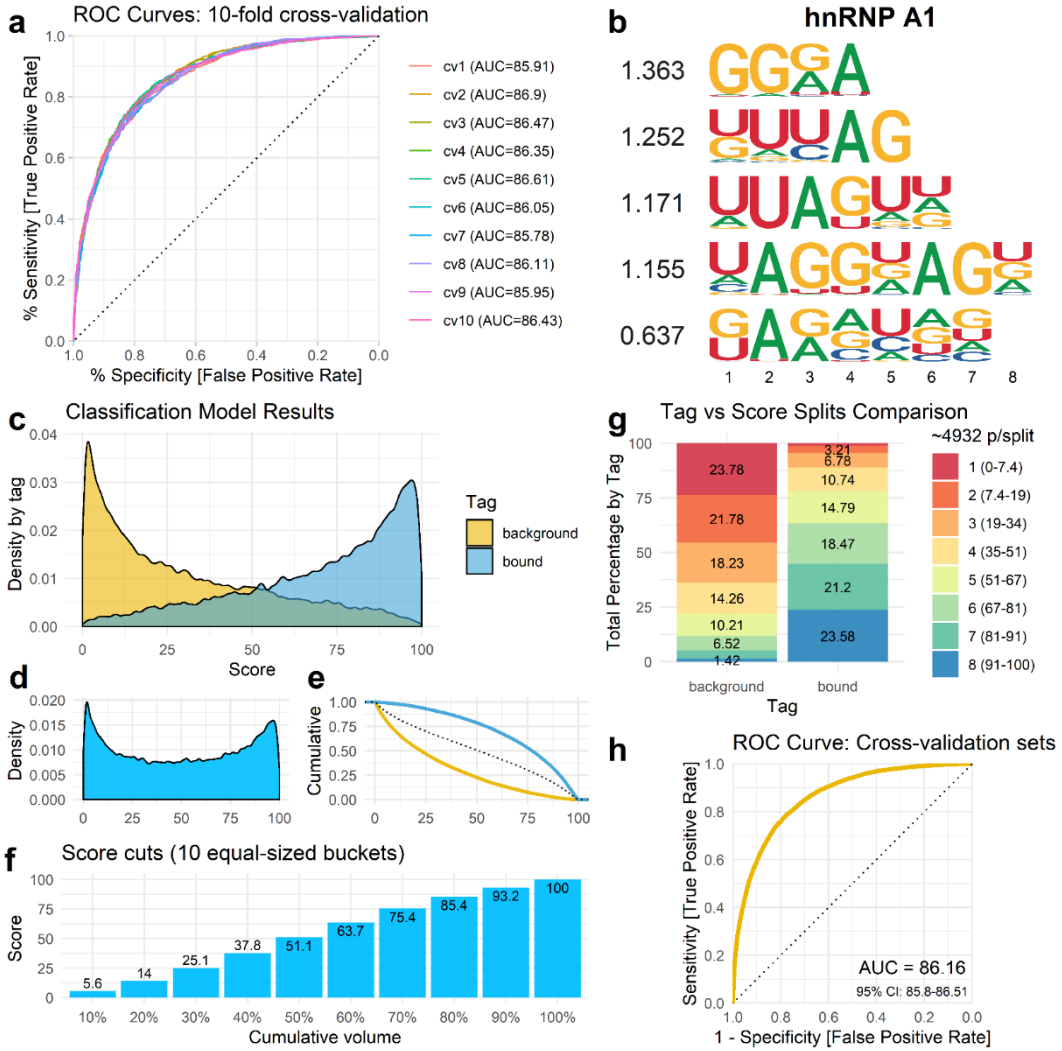


Figure S32 | DeepCLIP model characteristics for hnRNP A1. (a) Area under curve analysis of DeepCLIP models trained on hnRNP A1 iCLIP data from Bruun et al. using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

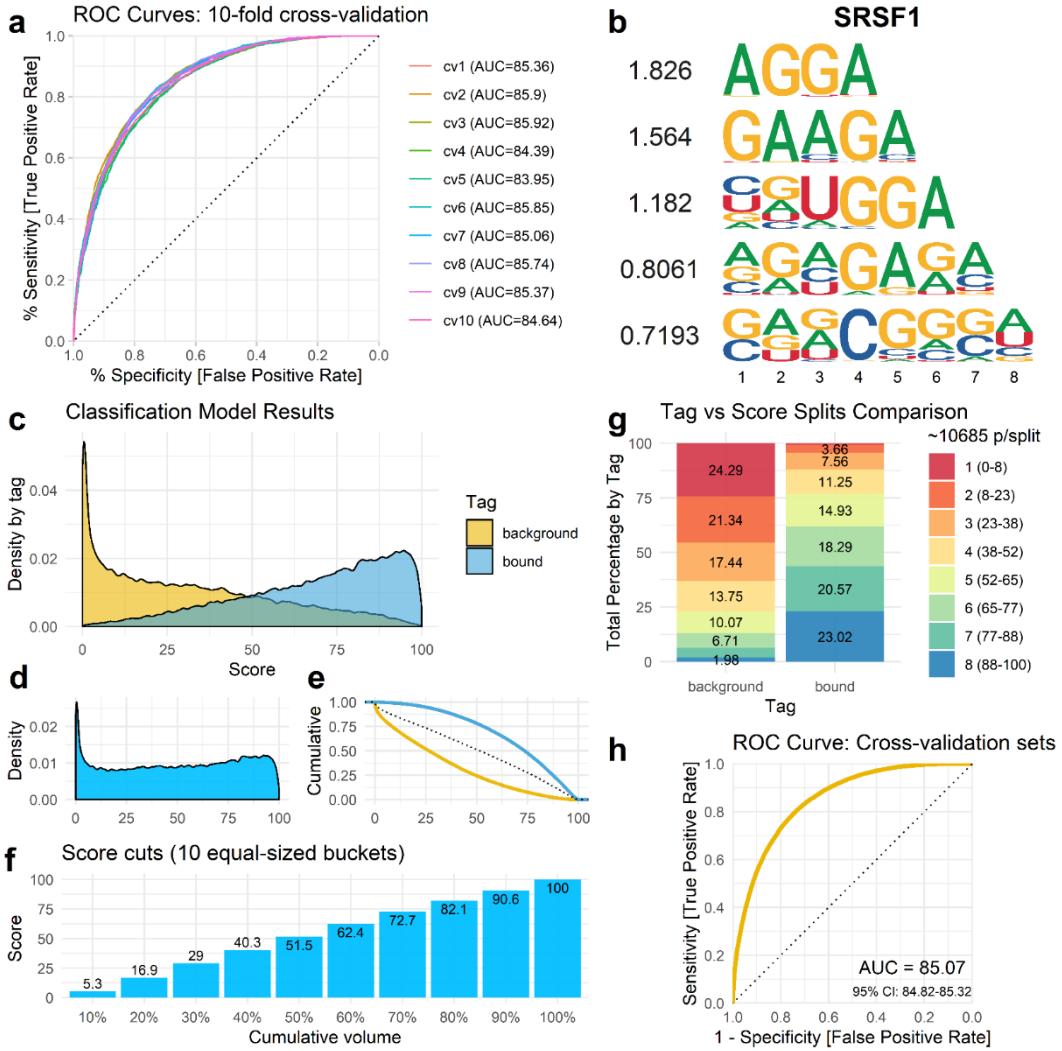


Figure S33 | DeepCLIP model characteristics for SRSF1 (eCLIP). (a) Area under curve analysis of DeepCLIP models trained on SRSF1 eCLIP data from Van Nostrand et al using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

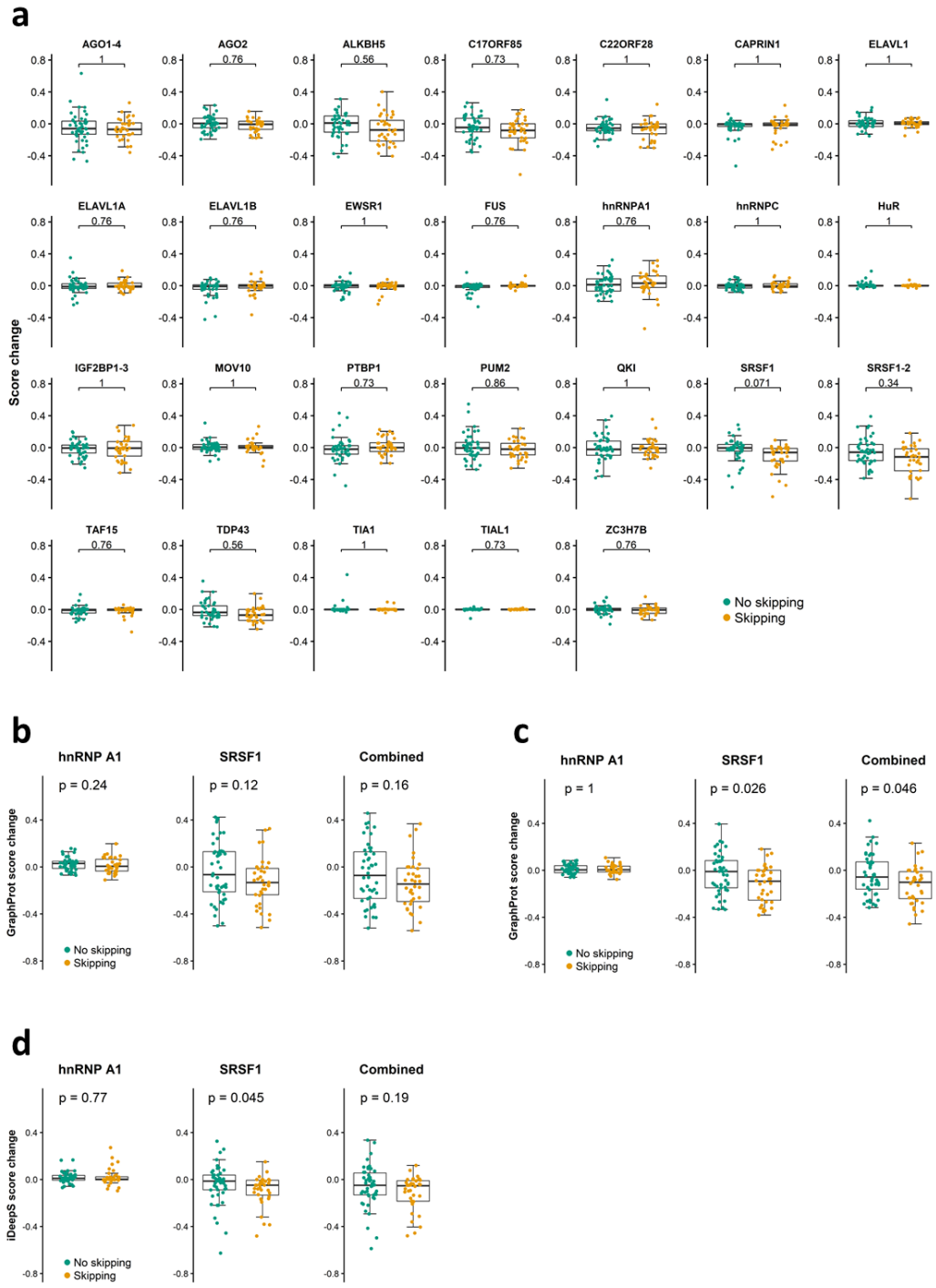


Figure S34 | Boxplots of Raponi dataset using previously trained models. (a) Boxplots of Raponi dataset using all models in the GraphProt benchmark dataset, and the new SRSF1 and hnRNP A1 model used in this study. B) Analysis of Raponi et al dataset with GraphProt models trained on the SRSF1 eCLIP and hnRNP A1 iCLIP data used to train DeepCLIP models in this study., padded with 150 nt genomic context on each side, according to GraphProt guidelines.. Input sequences for Raponi et al analysis were 15 nt padded with 150 nt genomic sequence on each side of the 15 nt viewpoint. (c) Same as (b), but with 75nt viewpoint. (d) Analysis of Raponi et al dataset with iDeepS models generated from 101nt input training data corresponding to the same sites used to train SRSF1 and hnRNP A1 DeepCLIP and GraphProt models. In all charts are shown boxplot of the score change between wt and mutation leading either to skipping (yellow) or no skipping (green).

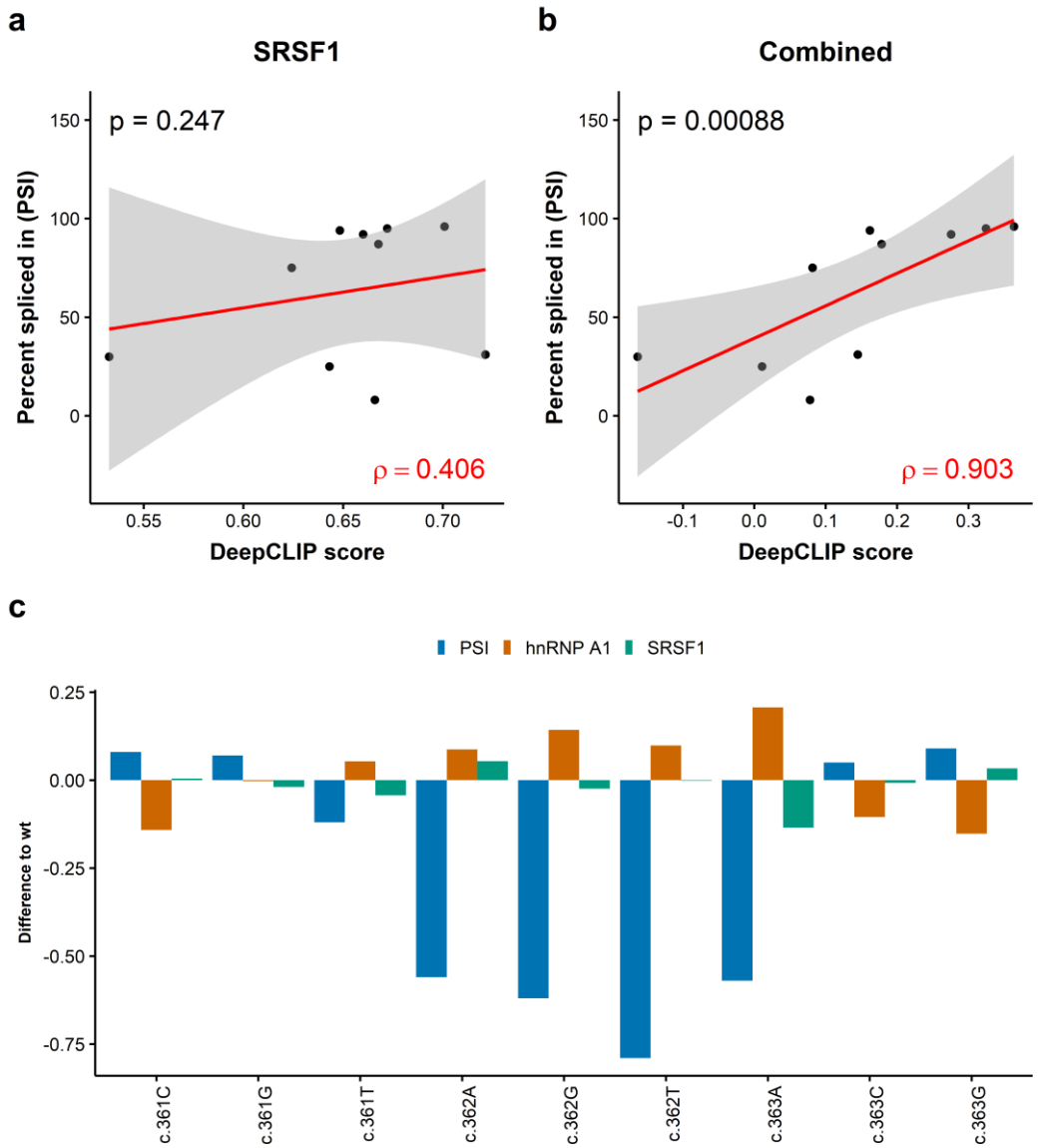


Figure S35 | DeepCLIP predictions of ACADM exon 5 mutations using SRSF1 (HITS-CLIP) model. (a) Scatter plot of ACADM exon 5 minigene percent spliced in (PSI) values and DeepCLIP SRSF1 score with linear regression (red line) and 95% confidence interval (shaded area). (b) Same as (a) but with the DeepCLIP hnRNP A1 score subtracted from the SRSF1 score. Spearman's rho is indicated in red for both plots in (a-b). (c) Barplot of the change relative to the wt of exon inclusion levels (blue), hnRNP A1 DeepCLIP scores (brown) and SRSF1 HITS-CLIP DeepCLIP model scores (green). The hnRNP A1 results are the same as presented in figure 5 but are included here for reference.

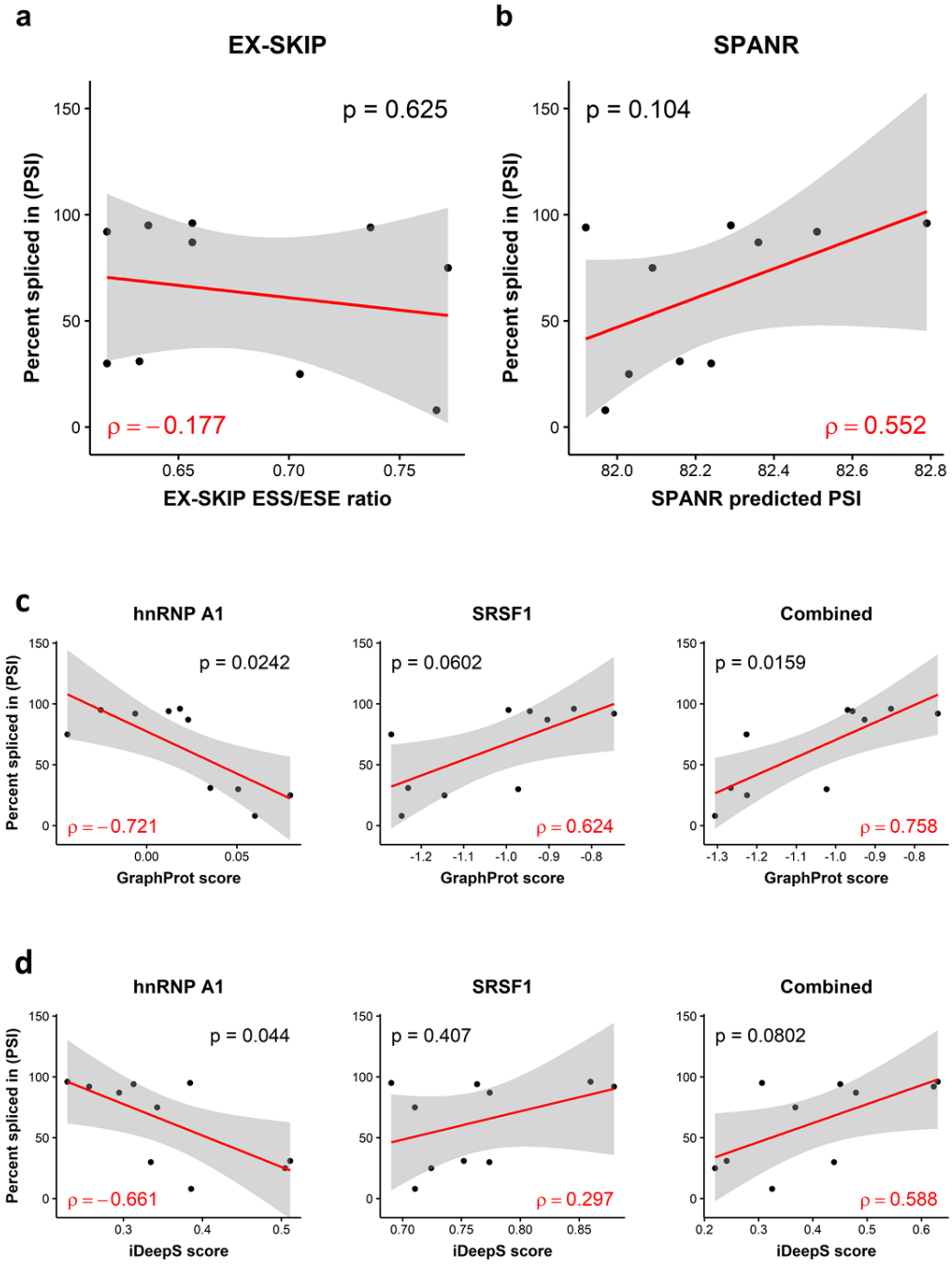


Figure S36 | Correlation between observed exon skipping and exon skipping model predictions. (a) Scatter plot of exon inclusion levels vs EX-SKIP predictions. (b) Scatter plot of the exon inclusions levels vs the SPANR prediction exon inclusion levels. (c) Scatter plots of exon inclusion levels vs predictions of GraphProt hnRNP A1 model (left), GraphProt SRSF1 model (middle) and GraphProt hnRNP A1 and SRSF1 models combined (right). The combined change in GraphProt scores is obtained by subtracting the hnRNP A1 scores from the SRSF1 scores (d) same as (c) but with iDeepS hnRNP A1 and SRSF1 models. Spearman's rho is indicated in red for both plots in (a-d).

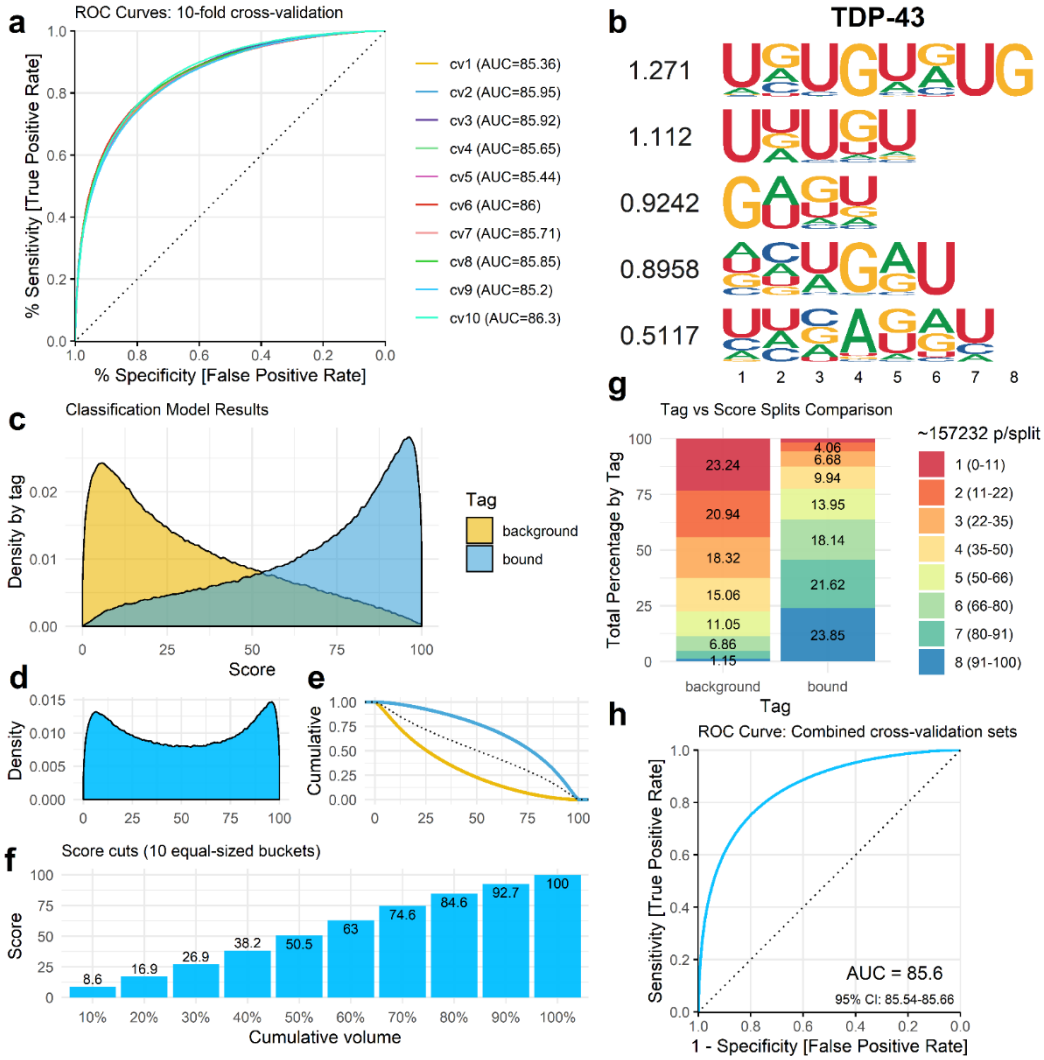


Figure S37 | DeepCLIP model characteristics for TDP43. (a) Area under curve analysis of DeepCLIP models trained on TDP43 binding sites from the POSTAR2database using 10-fold cross-validation. (b) Visualization of the CNN filters learned by the best performing model based on AUC. Score is equal to the mean information per base. (c-h) Visualizations of the combined model predictions of the 10-fold cross-validation. Scores are scaled to 0-100. (c) Density of background and bound prediction scores. (d) Combined density of prediction scores. (e) Cumulative predictive score of background and bound input sequences. (f) Barplot of cumulative scores of all input sequences. (g) Split prediction scores of background and bound input sequences. (h) Combined AUROC analysis using the pROC R package with DeLong estimation of 95% confidence interval.

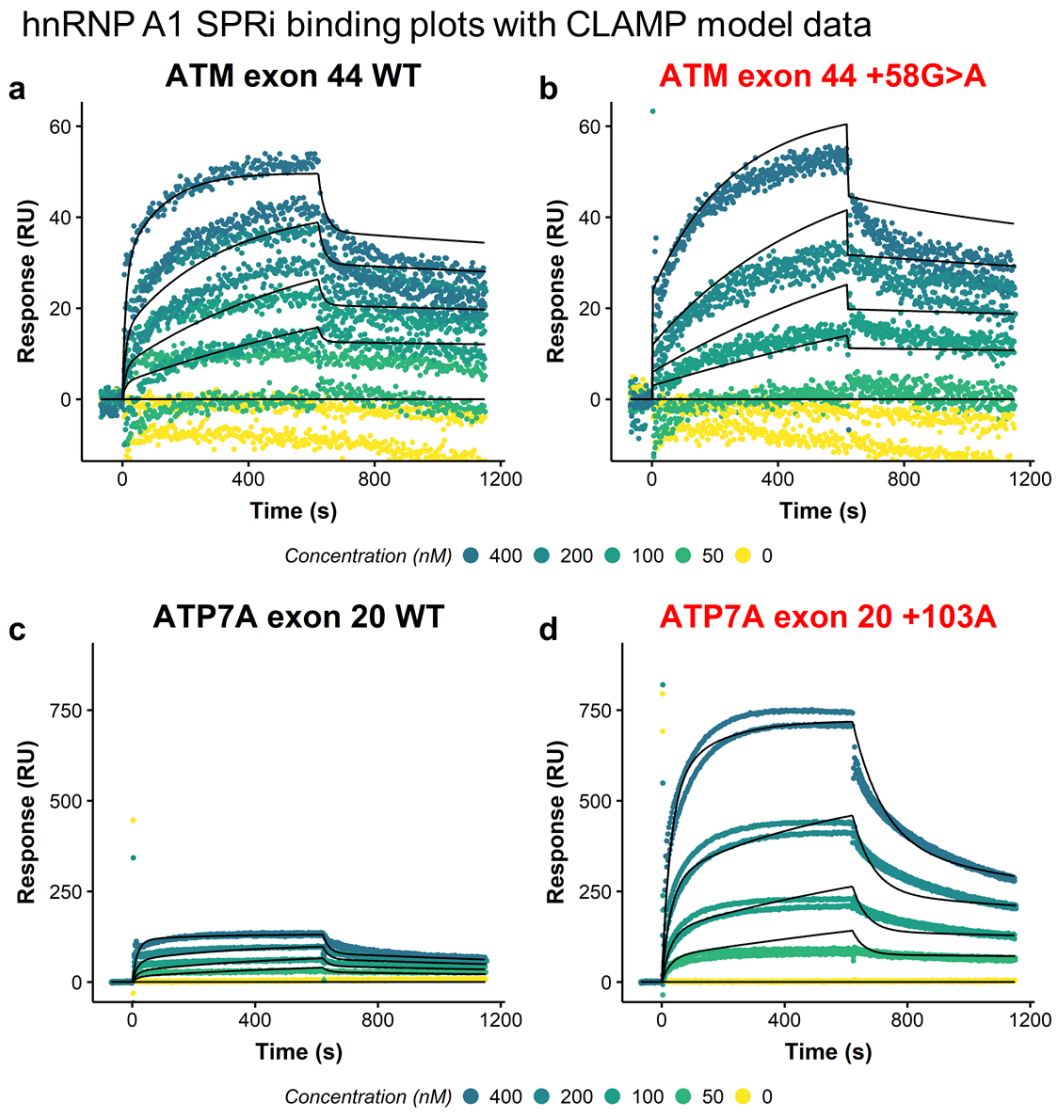


Figure S38 | Results of SPRi measurements of hnRNP A1 binding to a set of RNA wt and mutant oligos. Plots showing measured response (RU) versus time of the wt oligo (left) and mutant (right). The combined model fit across concentrations is indicated in black and concentrations are shown in decreasing order as gradually changing colors from blue through green to yellow. The fitted model's maximum simulated value is indicated above each plot.

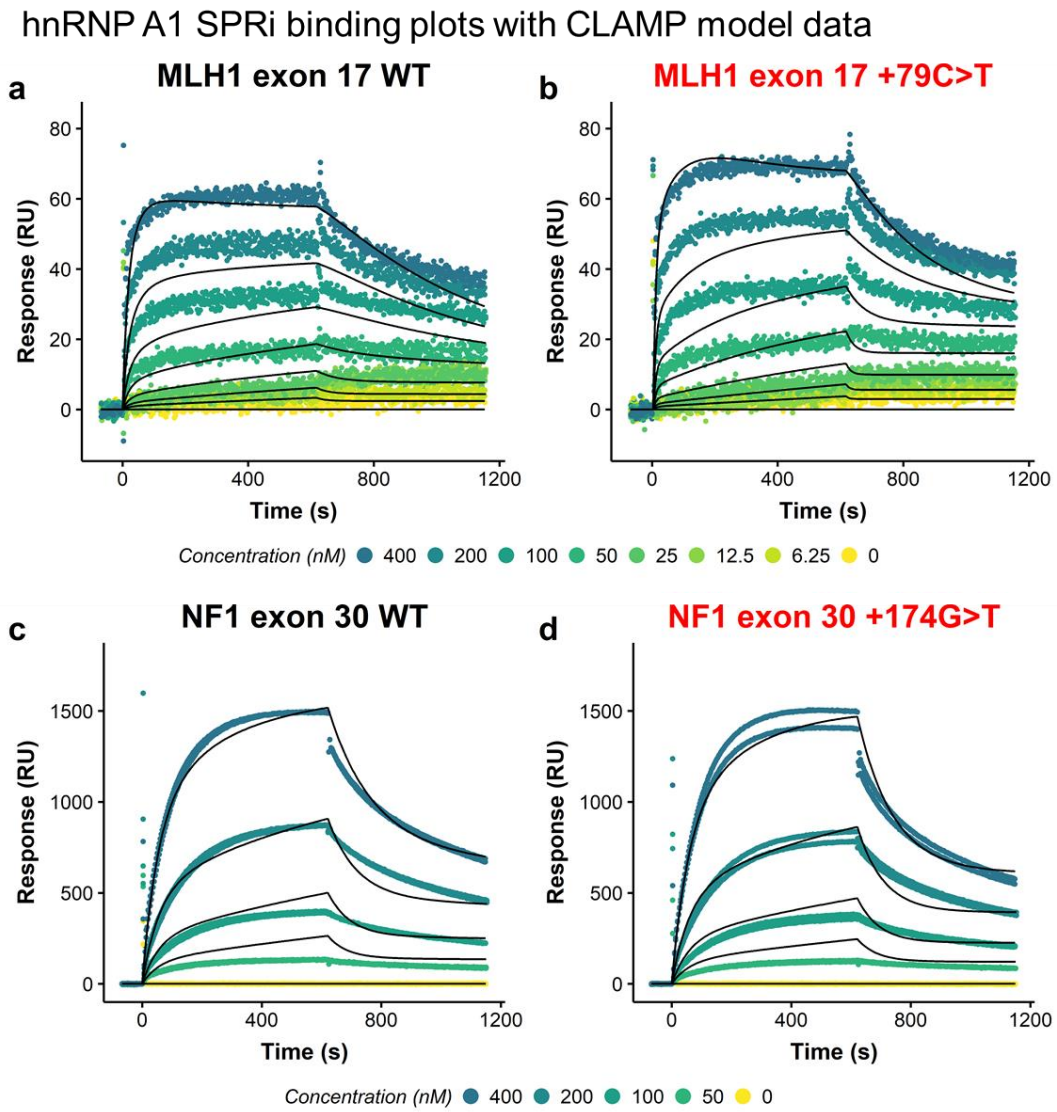


Figure S39 | Results of SPRi measurements of hnRNP A1 binding to a set of RNA wt and mutant oligos. Plots showing measured response (RU) versus time of the wt oligo (left) and mutant (right). The combined model fit across concentrations is indicated in black and concentrations are shown in decreasing order as gradually changing colors from blue through green to yellow. The fitted model's maximum simulated value is indicated above each plot.

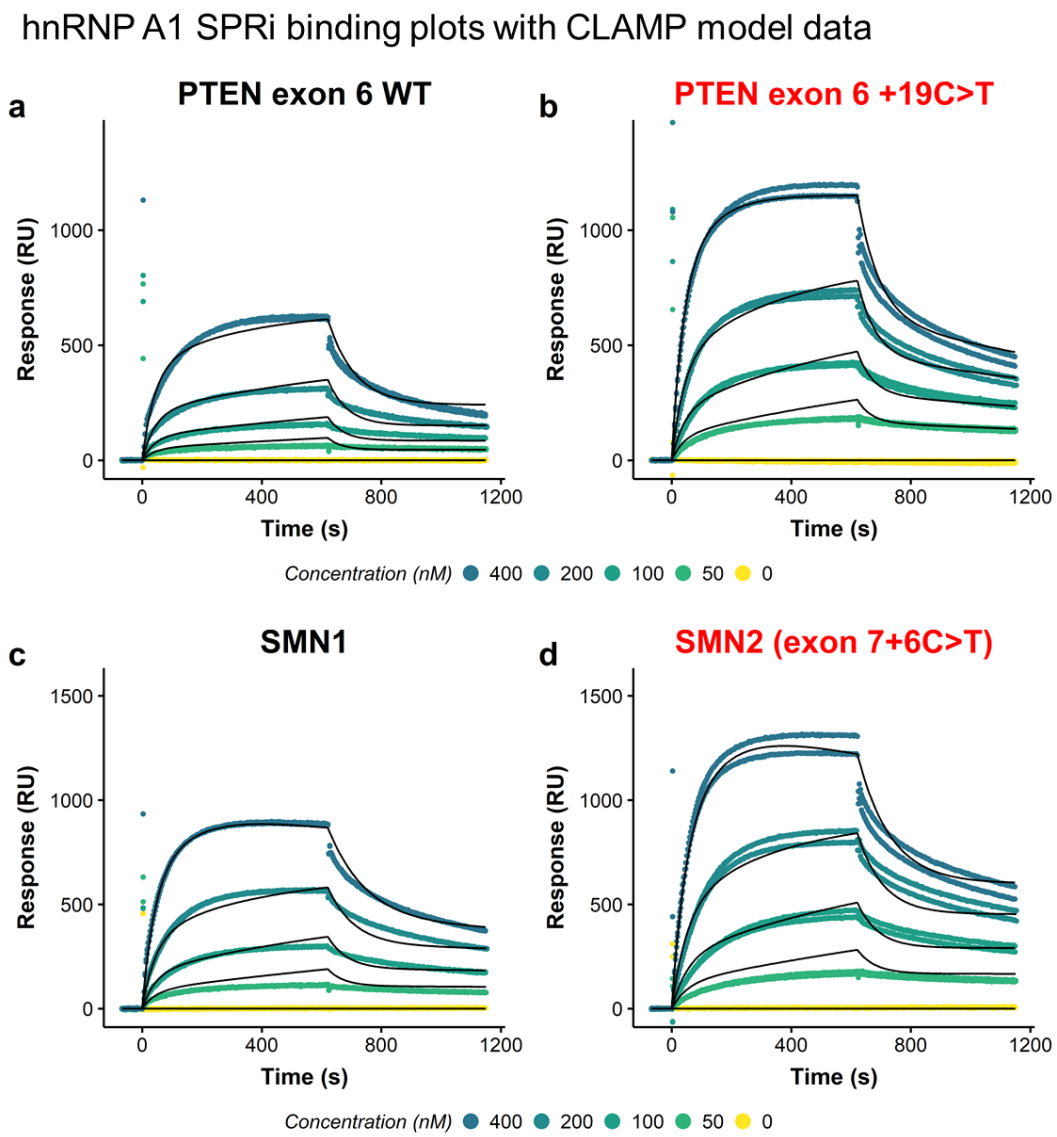


Figure S40 | Results of SPRi measurements of hnRNP A1 binding to a set of RNA wt and mutant oligos. Plots showing measured response (RU) versus time of the wt oligo (left) and mutant (right). The combined model fit across concentrations is indicated in black and concentrations are shown in decreasing order as gradually changing colors from blue through green to yellow. The fitted model's maximum simulated value is indicated above each plot.

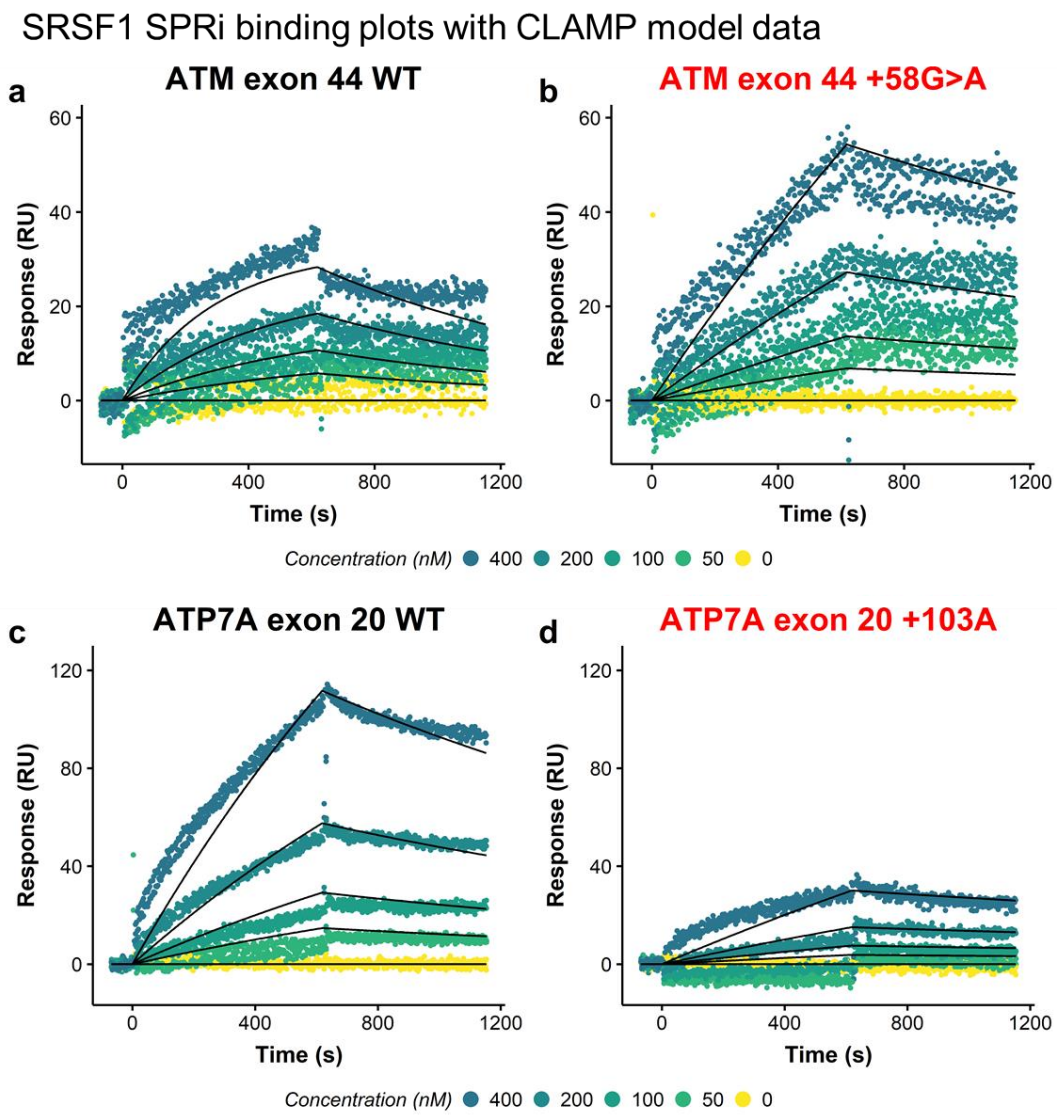


Figure S41 | Results of SPRi measurements of SRSF1 binding to a set of RNA wt and mutant oligos. Plots showing measured response (RU) versus time of the wt oligo (left) and mutant (right). The combined model fit across concentrations is indicated in black and concentrations are shown in decreasing order as gradually changing colors from blue through green to yellow. The fitted model's maximum simulated value is indicated above each plot.

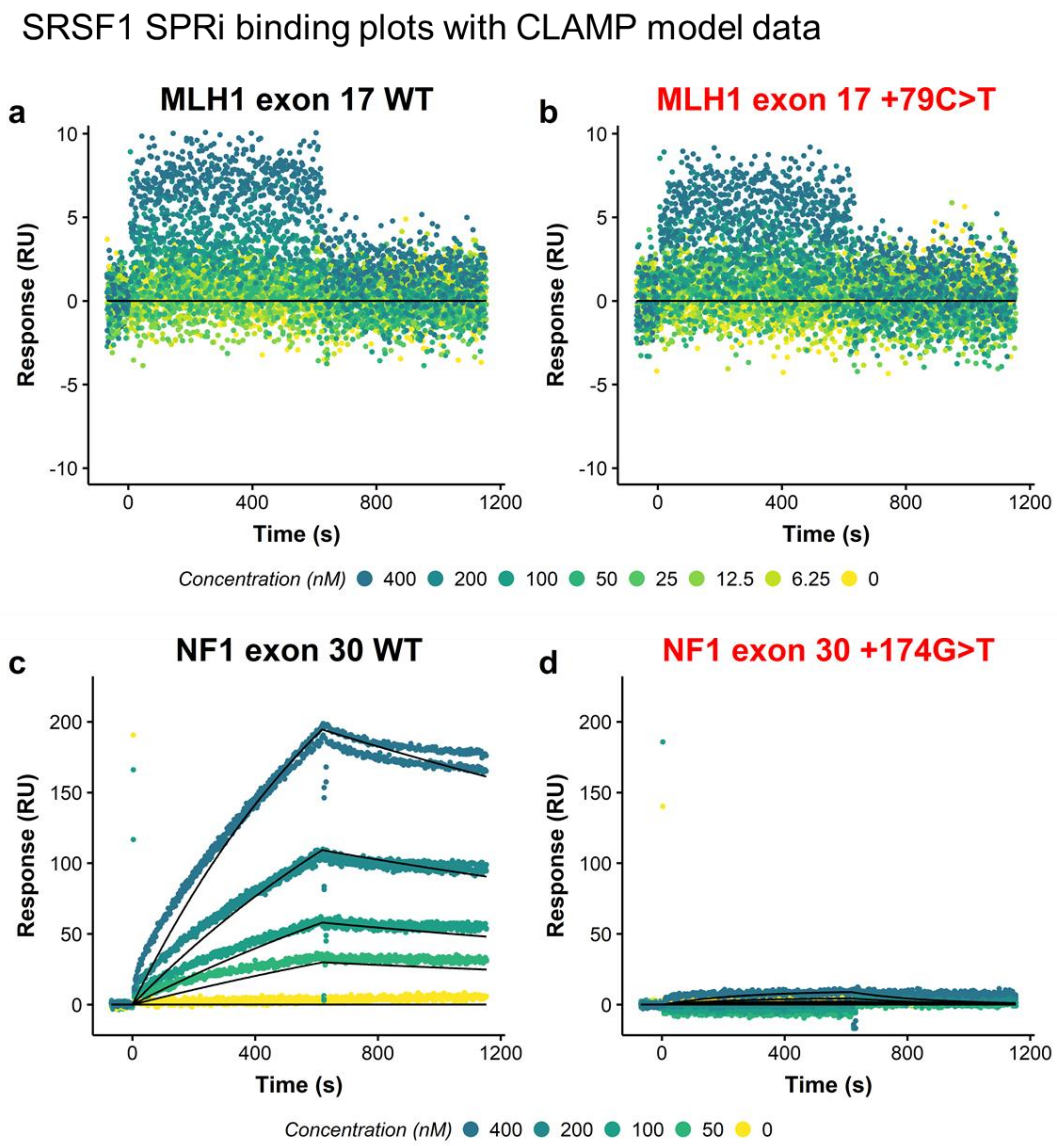


Figure S42 | Results of SPRi measurements of SRSF1 binding to a set of RNA wt and mutant oligos. Plots showing measured response (RU) versus time of the wt oligo (left) and mutant (right). The combined model fit across concentrations is indicated in black and concentrations are shown in decreasing order as gradually changing colors from blue through green to yellow. The fitted model's maximum simulated value is indicated above each plot.

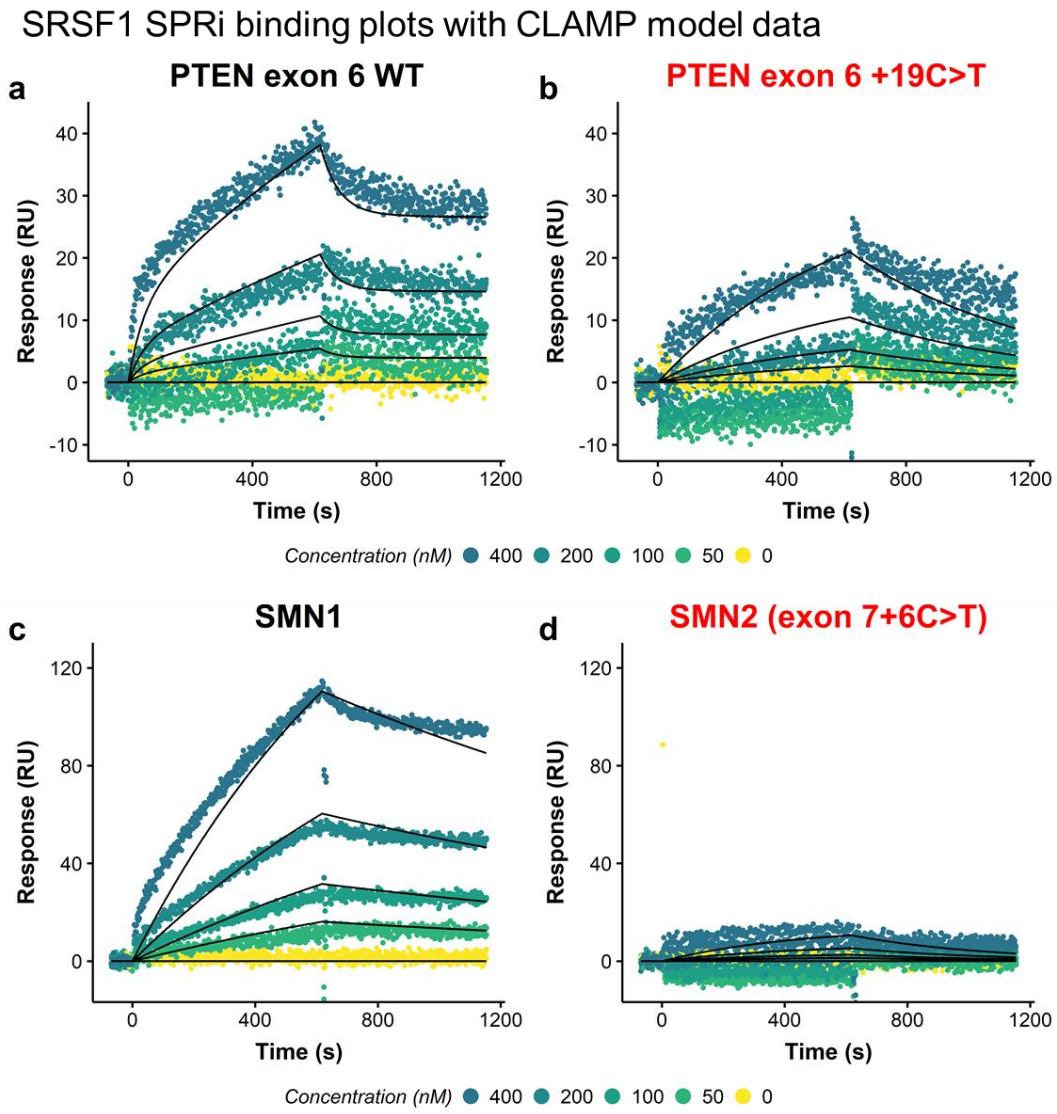


Figure S43 | Results of SPRi measurements of SRSF1 binding to a set of RNA wt and mutant oligos. Plots showing measured response (RU) versus time of the wt oligo (left) and mutant (right). The combined model fit across concentrations is indicated in black and concentrations are shown in decreasing order as gradually changing colors from blue through green to yellow. The fitted model's maximum simulated value is indicated above each plot.

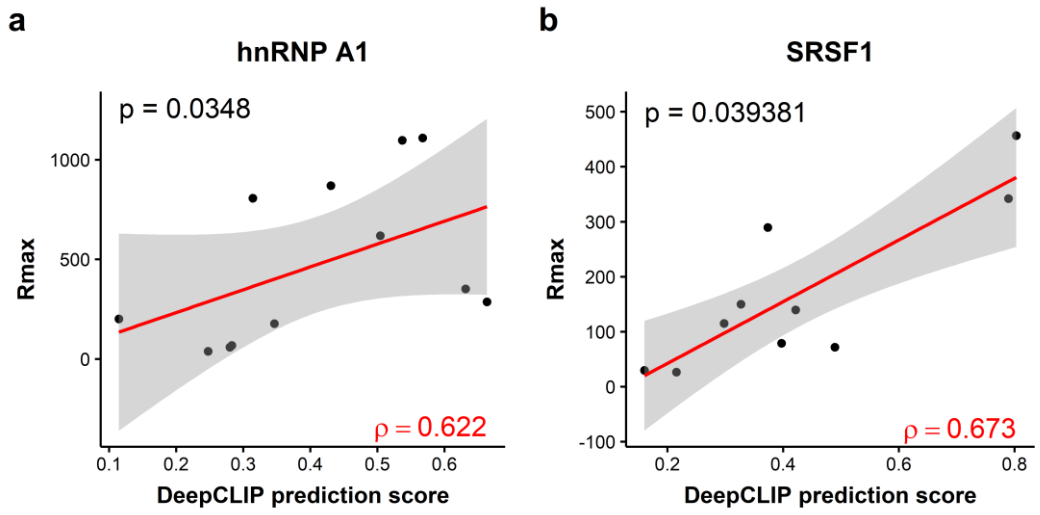


Figure S44 | Correlation between observed in vitro binding and DeepCLIP hnRNP A1 and SRSF1 model predictions. (a) Scatter plot of Rmax binding values from SPRi measurements of hnRNP A1 binding to oligos vs the predicted DeepCLIP hnRNP A1 scores. (b) Scatter plot of Rmax binding values from SPRi measurements of SRSF1 binding to oligos vs the predicted DeepCLIP SRSF1 scores. Spearman's rho is indicated in red in lower right corner and p-value in red in upper left corner for both plots in (a-b).

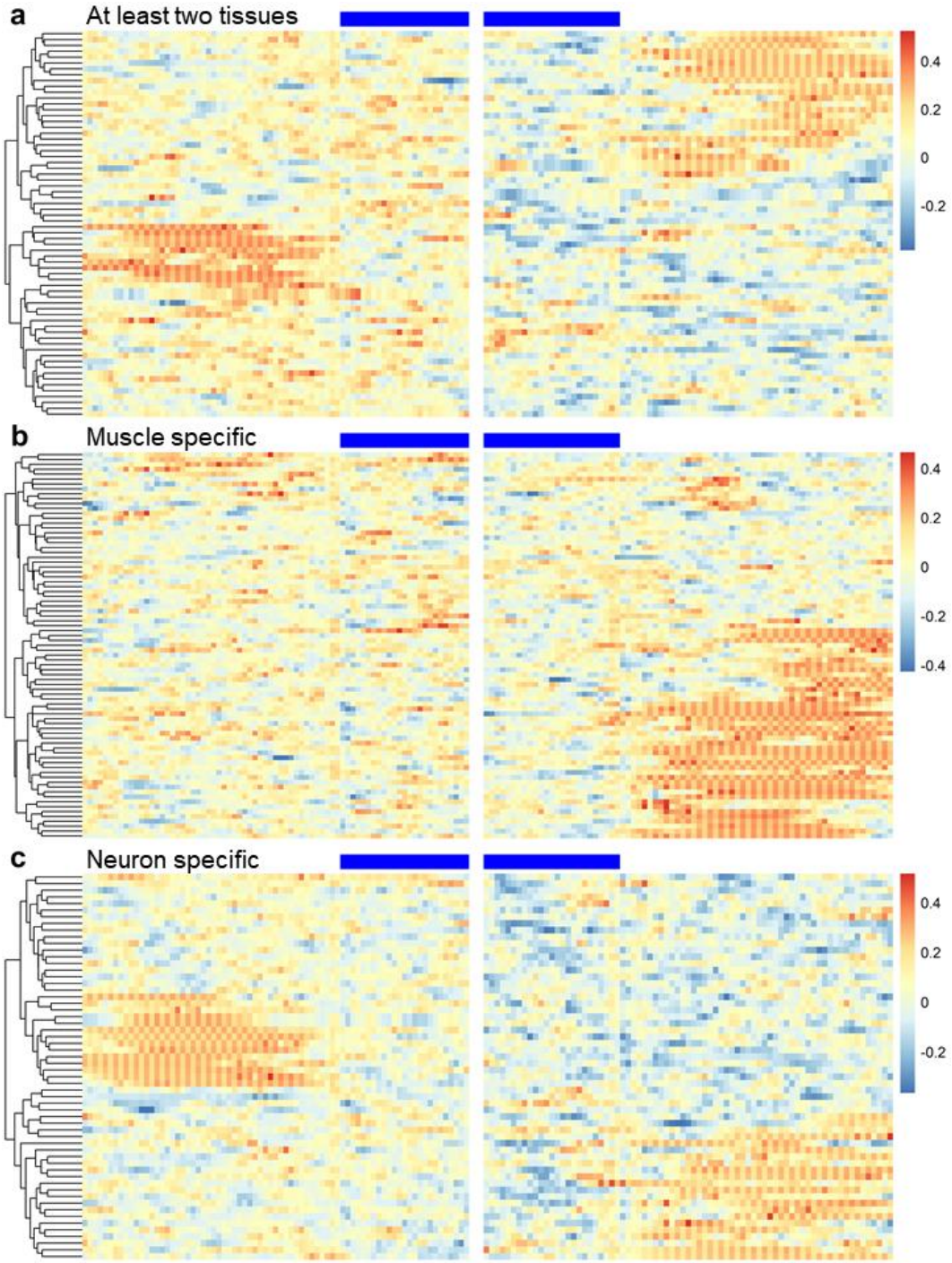


Figure S45 | DeepCLIP predicted binding at TDP-43 repressed pseudoexons. (a-c) Heatmap of TDP-43 DeepCLIP binding profiles at the 3'ss and 5'ss regions of TDP-43 repressed pseudoexons in mice. In all heatmaps the plots show 25 nt into the exon indicated by blue bars at the top, and the 50 first and last nt of the introns. The heatmap in (a) shows pseudoexons up-relugated upon TDP-43 depletion in at least two of the examined tissues (stem cells, neurons, and muscle cells), while the heatmap in (b) show muscle specific pseudoexons and the heatmap in (c) shows neuron specific pseudoexons.

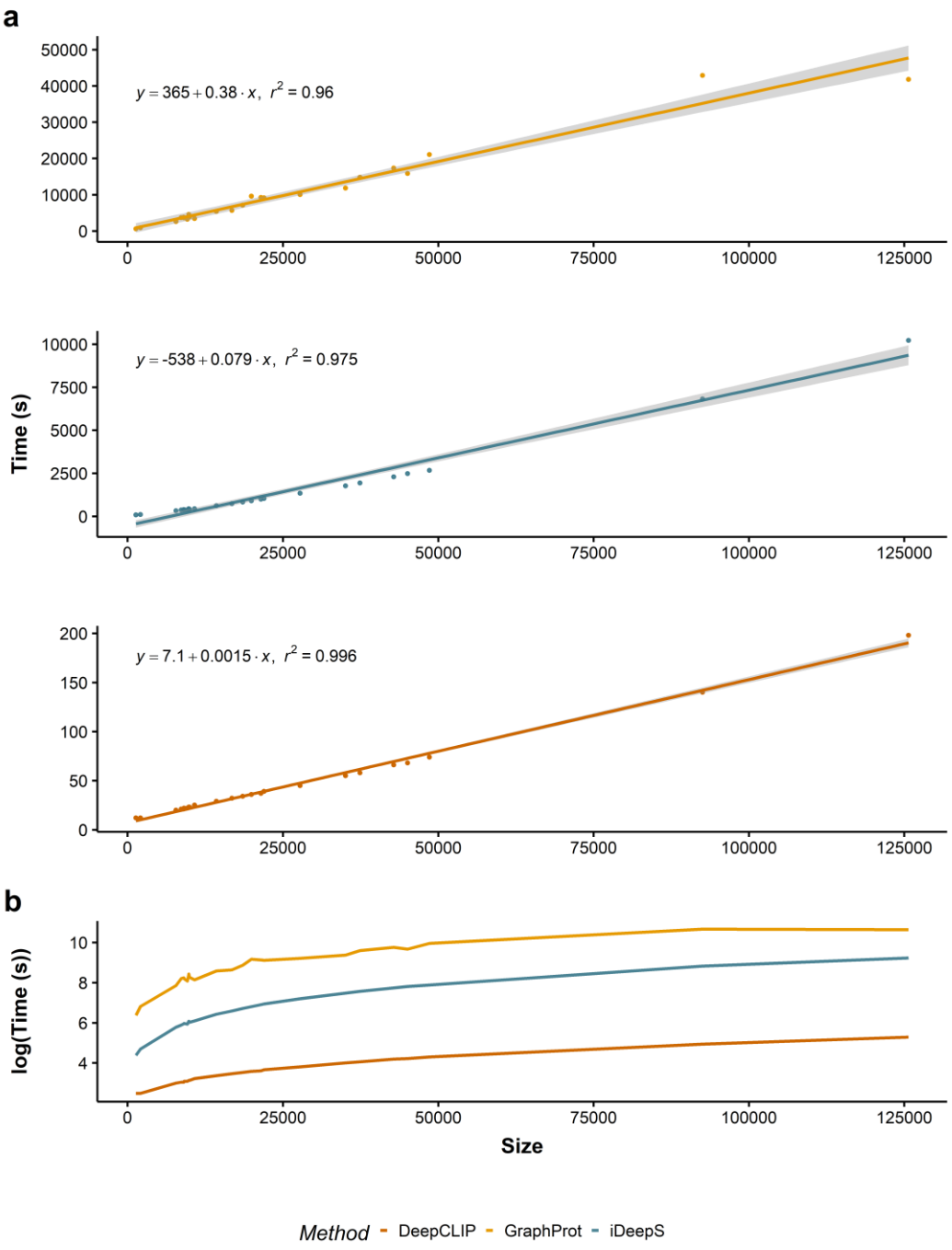


Figure S46 | Comparison of runtime lengths between GraphProt, iDeepS, and DeepCLIP. (a) Linear regression analysis of runtime lengths using the GraphProt benchmark dataset, each model timed on its own positive training data. GraphProt results are shown at the top, iDeepS in the middle and DeepCLIP at the bottom. The results of linear regression are indicated in each plot. (b) Log-linear plot of the data in (a) to compare the three methods on an equal scale. All methods were run on an Linux machine with an Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz with 32G RAM and Tesla K40c GPU.

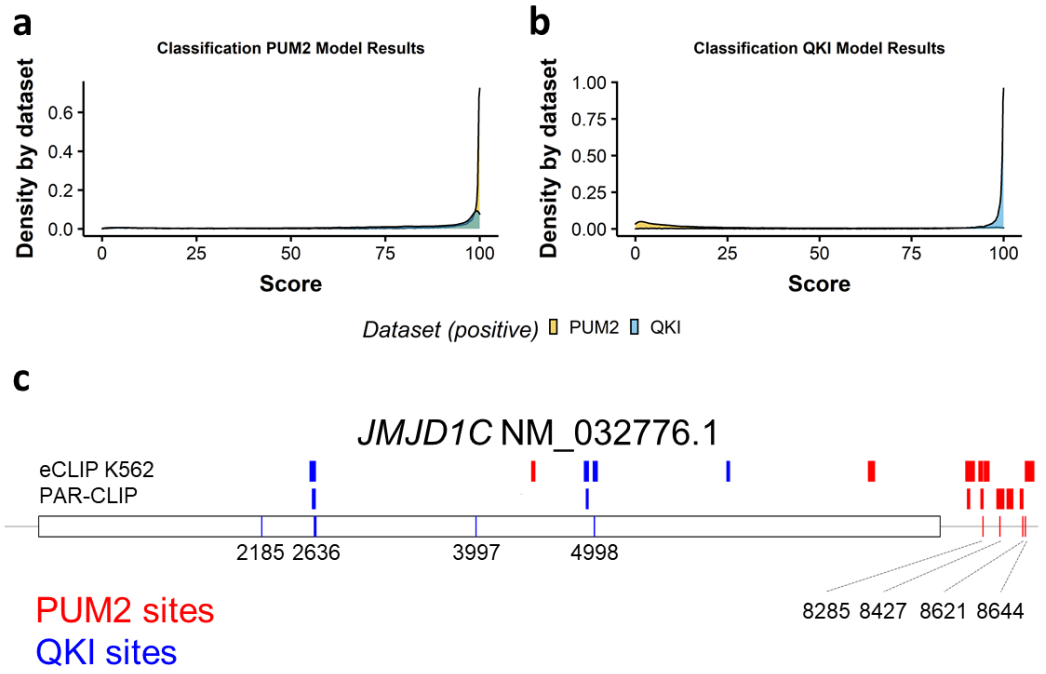


Figure S47 | Benchmark of PUM2 and QKI models. (a) The DeepCLIP PUM2 model was used to score the positive PUM2 dataset (yellow) and the positive QKI dataset (blue) and the distribution of scores was plotted as a density profile. (b) Same as (a), but using the DeepCLIP QKI model to score the two datasets. (c) The PUM2 model and QKI model were used in long-prediction mode to produce binding profiles across the length of the NM032776.1 transcript (8762 nt total). Predicted binding sites for each model were obtained by finding windows of at least 9nt length with a mean score of at least 0.3. PAR-CLIP and eCLOP sites are indicated above the transcript and DeepCLIP predicted sites within the transcript. The coding sequence of the transcript is indicated by a bar, and UTR regions by lines.

8 Supplementary materials for Manuscript 3

Table S1 | All found PKG-I targets. RHOA is not in the final list used for the simulations.

All found PKG-I targets
SRF
KCNQ2
CREB1
CFTR
LIPE
PRKCE
HSPB1
GATA4
TBXA2R
PDE5A
GTF2I
MAPK14
CBS
PRKCA
KCNQ3
ACTA1
GJD2
SMAD4
MRVI1
SPP1
SF1
RHOA

Table S2 | Log2 difference in gene expression of NOX-TFs from simulations based on random start distributions.

Run 1 (seed 111)	
E2F1	-3.1006442029646157
JUN	7.345194627281265
NFKB1	-1.4572010061164802
RELA	-1.6389319933193953
SP3	0.27770753447945934
STAT1	1.1491946451928106
STAT3	-1.681879908539212
Run 2 (seed 1111)	
E2F1	-2.7443325778887835
JUN	7.346156390310058
NFKB1	-1.707752504002778
RELA	-1.8501276053542903
SP3	0.2946143385903905
STAT1	0.8646549824830685
STAT3	-1.6131219836182769
Run 3 (seed 111124)	
E2F1	-3.0723369612790052
JUN	7.936589247077096
NFKB1	-1.4769074298904314
RELA	-1.9892323641417766
SP3	0.20295384570859495
STAT1	0.8522860232166909
STAT3	-1.574897925596644
Run 4 (seed 1134)	
E2F1	-2.9741871403014204
JUN	7.147834379033265
NFKB1	-1.2698130788679067
RELA	-1.898019595328511
SP3	0.28605547873193504
STAT1	0.9875567785943612
STAT3	-1.669855427397645
Run 5 (seed 91919)	
E2F1	-2.7538402698048676
JUN	7.877961503010229
NFKB1	-1.9317689821707371
RELA	-2.093513571693056
SP3	0.28545446884383824
STAT1	1.0513530133737277
STAT3	-1.54322409030177

9 Valorization

The continuous production and accumulation of biological and biomedical data have led the scientific community into a "big data era". In this era, data and computers have become one of our first-line defenses against disease development and progression. With the vast amount of data comes new opportunities and possibilities, which have motivated the construction of novel powerful analytical and machine learning algorithms that can locate and extract key patterns from the various and large data sets. Already, the marriage of big data and machine learning is used for diagnostic and prognostic purposes in clinical settings, and to identify new drug targets and purposes of validated drugs. Thus, the continuation of efficient data handling and the development of algorithmic analytical models are vital for the perpetual battle against diseases, disorders and illnesses.

Data mining and algorithmic analyses of biological and biomedical data is typically carried out by bioinformaticians, who combine computer science, mathematics, statistics and biological and biomedical knowledge to discover new insights and previously hidden patterns. It has now become evident that bioinformaticians are playing and will play a key role in carrying the weight of the opportunities, possibilities and responsibilities that have emerged with the large amounts of data produced in recent years. Now more than ever is the collaboration amongst and across scientific fields essential, as the bioinformaticians' classifiers, clustering algorithms and simulations always need guidance from biologists and medical staff whose expert knowledge serves as important feedback for the ongoing endeavor of improving the data driven models and their predictive capabilities. Such (direct and indirect) collaborations are in fact part of an already existing loop that (roughly) goes from biological data → bioinformatic models and predictions → "wet lab" experiments → biological data → … and so on. This loop ensures that both experimental data and bioinformatic solutions keep refining each other, which synergistically pave the way for new insights, discoveries and innovations.

Alongside the above-described "knowledge-loop" that strongly influence the design of current biomedical hypotheses and approaches, an interesting and significant scientific movement is beginning to pick up momentum. The expanding terrain of new biomedical data and novel algorithmic approaches is enabling researchers to see biological phenomena in a new and more clear light and to find more detailed explanations of observed molecular events. A pivotal example of this tendency is the changing conceptual understanding of diseases and their definitions. The current disease definition is one based on symptoms, however, with the growing amount of data, knowledge and insights scientists and clinicians are beginning to dig deeper for satisfying explanations and nosological connections. And it is getting clearer that diseases are more meaningfully defined in terms of causal mechanisms instead of the symptoms they generate, as the same symptoms can be produced by distinct disease-causing mechanisms. A bonus of such a redefinition of the disease term is that the specific and underlying cause is already identified, which again

readily reveals how to treat it via e.g. personalized medicine and network pharmacology approaches. Thus, bioinformatic work is an important link in the chain of our scientific evolution, as the bioinformatic solutions are key factors in driving the progress and for keeping the momentum of the slowly but steadily accelerating scientific movement.

This thesis presented three manuscripts that each either indirectly or directly showed how a synergetic use of bioinformatic solutions and experimental data can lift the impact of scientific discoveries to new heights. Additionally, all the manuscripts revealed findings that can aid a general mechanism-based disease understanding and therapy development.

Starting with the first included manuscript (chapter 2), here we presented a new clustering method for scRNA-seq data that can analyze and compare single-cell development trajectories. For data from studies on healthy cell differentiations and disease progression, we showed that our innovative method can identify mechanistic patterns involved in driving the developments of the cells. We hypothesize that our tool's findings can serve as key steps in drug repurposing pipelines and perhaps help formulate mechanism-based disease definitions. For instance, when we analyzed the scRNA-seq data from a study on CD8 T-cell development in chronic infections, we located potential drug targets that may help establish new hypotheses about therapy development.

In the second manuscript (chapter 3), we introduced a new simple neural network classifier and demonstrated that the network can reveal how nucleotide variants affect binding affinity of RNA-binding proteins. The predictions and binding profiles produced by our tool directly relates to the discovery of mechanisms underlying diseases. And as a new feature, we showed how our models can be used to guide the development of SSO-therapies (splice-switching oligonucleotide therapies), which might be a valuable technique in clinical settings and for the medicinal industry. A known disease that is treated with SSO-therapy is Spinal Muscular Atrophy (SMA). We are convinced that it is only a matter of time before many similar treatment approaches will be invented and pass the clinical trials.

The third manuscript (chapter 4) described a simulation approach that was invented with the sole purpose of unraveling mechanisms underlying a detected PKG-dependent downregulation of ROS-producing proteins. And as it was suggested in the manuscript, the discovery of such a mechanism may be crucial to the progress of diagnosis and treatments for ischemic stroke induced damages. Both PKG and ROS signaling are represented by drugs in clinical practice. Several more applications are possible and being tested; some have already failed. However, it is highly likely that this was due to the fact that patients were included solely based on clinical parameters and not on mechanistic ones. Precision medicine needs both precision diagnostics and precision mechanism-based drugs. With the here generated tools, the search for new mechanisms and mechanism-based diagnostics

may be furthered. Nevertheless, further experimental analyses and clinical validation are needed before anything can be refined and/or concluded.

As it can be seen, this thesis is a child of its “scientific time”, as it addresses present and ongoing challenges, and is interlinked with the general and established knowledge-refining feedback loop and the scientific movement that the big data era has helped to bring about. Altogether, this thesis bears relevance for biology, biomedicine, bioinformatics and treatment development.

10 Appendix

10.1 List of Figures

Figure 1. Central dogma of molecular biology in eukaryotic cells. Starting from the top, DNA can replicate itself and be transcribed into RNA. RNA can replicate itself and be translated into protein. In addition, RNA can be reversely transcribed into DNA. DNA and RNA replication and transcription takes place in the nucleus, whereas translation takes place in the cytoplasm. Deoxyribonucleic acid, DNA; ribonucleic acid, RNA. 4

Figure 2. Splice sites and branch point. The figure only shows the most conserved splice site motifs [13]. 6

Figure 3. Position frequency matrix and sequence logo. 8

Figure 4. Illustrations of different graph types. 12

Figure 5. Feedforward neural network graph. All nodes in the network are illustrated as circles. Arrows show the flow of information. At the top, the input layer is shown. The circles (the nodes) are not filled with color, which indicates that these are not “processing units” of the network. The input layer feeds input data to Hidden layer 1. All nodes of Hidden layer 1 receive all input data simultaneously. The same is true for Hidden layer 2, but here the layer receives input from Hidden layer 1. Hidden layer 2 passes its outputs to the Output layer. 15

Figure 6. Three regularly used activation functions. a Sigmoidal activation function. All inputs to the function are squashed into values between 0 and 1. b Hyperbolic tangent (tanh) activation function. All inputs are squashed into values between -1 and 1. c Rectified linear unit (ReLU) activation function. All negative input values to the function are converted to zeros and all positive input values remain unchanged. 17

Figure 7. Valid convolutions of the input vector x using kernel $\boldsymbol{w}$. Input vector **x** and kernel $\boldsymbol{w}$ are shown at the top. Below, the convolutional operations are shown as well as the computation of the output. At the bottom, the output vector $\boldsymbol{z}$ is shown. * = row vector of size 1×4. 18

Figure 8. Recurrent neural network. All nodes in the network are illustrated as circles. Arrows show the flow of information. At the top, the input layer is shown. The circles (the nodes) are not filled with color, which indicate that these are not “processing units” of the network. The input layer feeds input data to Hidden layer 1, which consists of recurrent nodes. All nodes of Hidden layer 1 receive all input data simultaneously. All nodes of Hidden layer 1 have cyclic connections and are connected to all nodes in the same layer. The same is true for Hidden layer 2, but here the layer receives input from Hidden layer 1. Hidden layer 2 passes its outputs to the Output layer. 20

Figure 9. LSTM memory block. i is the input gate, g is the modulatory gate, c is the cell state, f is the forget gate, o is the output gate, $\boldsymbol{x}t$ is the input at timestep t, $\boldsymbol{h}t$ is the output at timestep t, $\boldsymbol{h}t-1$ is the output at timestep $t-1$ and $\odot$ indicates the Hadamard product (element-wise multiplication). The Figure is inspired by Figure 2 in [65]. 22

Figure 10. Aim of gradient descent. $E(\theta)$ is an error function that has a global minimum at $(0,0)$. By taking the derivative of $E(\theta)$ it is possible to identify how θ should be changed such that the global minimum can be reached. If $E'\theta > 0$ then θ should be decreased, while θ should be increased if $E'\theta < 0$. 25

Figure 11. Scatter plot and associated distance matrix. a Scatter plot showing position of the point $0, 1, 2, 3$ and 4. b Distance matrix based on the points shown in a) calculated using Euclidean distance. 27

Figure 12. Dendrograms and illustrations of single, average and complete linkage. a-c (left) dendrograms found using simple, average and complete linkage, respectively. The red horizontal bars match merge-distances of the single linkage dendrogram and they make it easier to see the different linkage types' impact on the resulting dendrogram. a-c (right) Illustrations of single, average and complete linkage. All dendrograms are calculated using the data points shown in Figure 11a. 29

Figure 13. Scatter plot with connected data points, constrained distance matrix and complete linkage clustering with and without constraints. a Scatter plot with connected data points. b Distance matrix reflecting the distance between possible cluster merges. c Dendrogram of constrained agglomerative hierarchical clustering given the point in a and using the complete linkage. d Dendrogram of agglomerative hierarchical clustering given the point in a without the connections and using the complete linkage. 30

10.2 List of publications

Publications and drafts included in this thesis:

1. **Alexander G. B. Grønning**, Mhaned Oubounyt, Kristiyan Kanev, Jesper Beltoft Lund, Tim Kacprowski, Dietmar Zehn, Richard Rottger, Jan Baumbach (2020) "*Comparative single-cell trajectory network enrichment identifies pseudotemporal systems biology patterns in hematopoiesis and CD8 T-cell development*". (Submitted to BioRxiv).

2. **Alexander G. B. Grønning**, Thomas Koed Doktor, Simon Jonas Larsen, Gitte Hoffmann Bruun, Ulrika Simone Spangsberg Petersen, Jan Baumbach, Brage Storstein Andresen (2020) "*DeepCLIP: Predicting the effect of mutations on protein-RNA binding with Deep Learning*". (Published, Nucleic Acids Research).

3. **Alexander G. Grønning**, Jan Baumbach, Harald H.H.W. Schmidt, Ana I. Casas (2020) "*Unraveling of the mechanism behind PKG-dependent regulation of NOX4, 5 gene expression for improved ischemic stroke therapy*". (Draft)

Other contributions not included in the thesis:

- Cristian Nogales*, **Alexander G. B. Grønning***, Sepideh Sadegh, Jan Baumbach, Harald H.H.W. Schmidt (2020) "*Network medicine based unbiased disease modules for drug and diagnostic target identification in ROSopathies*", Handbook of Experimental Pharmacology – Reactive Oxygen Species: Sources, Targets and Therapeutic Implications. (Submitted)

- Elisa Anastasi, Ahmed A. Hassan, **Alexander G. B. Grønning**, Tim Kacprowski, Jan Baumbach, Harald HHW Schmidt, Anil Wipat, Emre Guney (2019) "*On the need to overcome inconsistent symptom-based disease classifications*". (Unpublished).

- Published Rebecca E. Tweedell, Dingyin Tao, Timothy Hamerly, Tanisha M. Robinson, Simon Larsen, **Alexander G. B. Grønning**, Alessandra M. Norris, Jonas G. King, Henry Chun Hin Law, Jan Baumbach, Elke S. Bergmann-Leitner, Rhoel R. Dinglasan (2019) "*The Selection of a Hepatocyte Cell Line Susceptible to Plasmodium falciparum Sporozoite Invasion That Is Associated With Expression of Glypican-3*", Fronties in Micriobiology.

10.3 List of abbreviations

3’ ss	3’-splice site
5’ ss	5’-splice site
A	Adenine
C	Cytosine
CLIP	Crosslinking and immunoprecipitation
DNA	Deoxyribonucleic acid
ds-cDNA	Double-stranded copy DNA
FACS	Fluorescence-activated cell sorting
G	Guanine
GRN	Gene regulatory network
LSTM	Long short-term memory
mRNA	Messenger RNA
PCA	Principal component analysis
PFM	Position frequency matrix
PPI	Protein-protein interaction
pre-mRNA	Precursor mRNA
PWM	Position weight matrix
RBP	RNA-binding protein
RNA	Ribonucleic acid
RNA-seq	RNA sequencing
RNN	Recurrent neural network
Scellnetor	Single-cell Network Profiler for Extraction of Systems Biology Patterns from scRNA-seq Trajectories
scRNA-seq	Single-cell RNA sequencing
SELEX	Systematic evolution of ligands by exponential enrichment
SRE	Splicing-regulatory element
SNP	Single nucleotide polymorphism
T	Thymine
tRNA	Transfer RNA
U	Uracil
UMAP	Uniform Manifold Approximation and Projection
Y2H	Yeast two-hybrid
TAP/MS	Tandem affinity purification followed by mass spectrometry