bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

GraphProt2: A novel deep learning-based method for predicting binding sites of RNA-binding proteins

Michael Uhl[1], Van Dinh Tran[1], Florian Heyl[1], Rolf Backofen[1,2,*]

1 Bioinformatics Group, Department of Computer Science, University of Freiburg, Germany
2 Signalling Research Centres BIOSS and CIBSS, University of Freiburg, Germany

*** Corresponding author**

Abstract

CLIP-seq is the state-of-the-art technique to experimentally determine transcriptome-wide binding sites of RNA-binding proteins (RBPs). However, it relies on gene expression which can be highly variable between conditions, and thus cannot provide a complete picture of the RBP binding landscape. This necessitates the use of computational methods to predict missing binding sites. Here we present GraphProt2, a computational RBP binding site prediction method based on graph convolutional neural networks (GCN). In contrast to current CNN methods, GraphProt2 supports variable length input as well as the possibility to accurately predict nucleotide-wise binding profiles. We demonstrate its superior performance compared to GraphProt and a CNN-based method on single as well as combined CLIP-seq datasets.

Introduction

RNA-binding proteins (RBPs) regulate many vital steps in the RNA life cycle, such as splicing, transport, stability, and translation [1]. Recent studies suggest a total number of more than 2,000 human RBPs, including 100s of unconventional RBPs, i.e., RBPs lacking known RNA-binding domains [2–4]. Numerous RBPs have been implicated in diseases like cancer, neurodegeneration, and genetic disorders [5–7], urging the need to speed up their functional characterization and shed light on their complex cellular interplay.

An important step to understand RBP function is to identify the precise RBP binding locations on regulated RNAs. In this regard, CLIP-seq (cross-linking and immunoprecipitation followed by next generation sequencing) [8] together with its popular modifications PAR-CLIP [9], iCLIP [10], and eCLIP [11] has become the state-of-the-art technique to experimentally determine transcriptome-wide binding sites of RBPs. A CLIP-seq experiment for a specific RBP results in a library of reads bound and protected by the RBP, making it possible to deduce its binding sites by mapping the reads back to the respective reference genome or transcriptome. In practice, computational analysis of CLIP-seq data has to be adapted for each CLIP-seq protocol [12]. Within the analysis, arguably the most critical part is the process of peak calling, i.e., to infer RBP binding sites from the mapped read profiles. Among the many existing peak callers, some popular tools are Piranha [13], CLIPper [14], PEAKachu [15], and PureCLIP [16].

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

While peak calling is essential to separate authentic binding sites from unspecific interactions and thus reduce the false positive rate, it cannot solve the problem of expression dependency. In order to detect RBP binding sites by CLIP-seq, the target RNA has to be expressed at a certain level in the experiment. Since gene expression naturally varies between conditions, CLIP-seq data cannot be used directly to make condition-independent binding assumptions on a transcriptome-wide scale. Doing so would only increase the false negative rate, i.e., marking all non-peak regions as non-binding, while in fact one cannot tell from the data since there is no evidence from the CLIP-seq experiment. Moreover, expression variation is especially high for lncRNAs, an abundant class of ncRNAs gaining more and more attention due to their diverse cellular roles [17]. It is therefore of great importance to infer RBP binding characteristics from CLIP-seq data in order to predict missing binding sites. To give an example, [18] investigated the role of the splicing factor PTBP1 in differential splicing of the tumor suppressor gene ANXA7 in glioblastoma. Despite strong biological evidence for PTBP1 directly binding ANXA7, no binding site was found in a publicly available CLIP-seq dataset for PTBP1. Instead, only a computational analysis was capable to detect and correctly localize the presence of potential binding sites which were then experimentally validated.

Over the years, many approaches to RBP binding site prediction have been presented, from simple sequence motif search to more sophisticated methods incorporating classical machine learning and lately also deep learning. Some popular earlier methods include RNAcontext [19] and GraphProt [20], which can both incorporate RNA structure information into their predictive models. While RNAcontext utilizes a motif model incorporating both sequence and structural context, GraphProt uses a graph kernel approach showing improved performance over motif-based techniques. From 2015 on, various deep learning based methods have been proposed, starting with DeepBind [21], which uses sequence information to train a convolutional neural network (CNN). Subsequent methods largely built upon this methodology, using CNNs in combination with recurrent neural networks and additional features such as structure, evolutionary conservation, or region type information [22]. While these methods certainly provide state-of-the-art predictive performance, CNNs by design restrict the input sequences to a fixed length for a given model. Moreover, they cannot encode base pair information, i.e., annotated connections between non-adjacent bases, calling for a more flexible approach that can deal with these limitations, while at the same time supporting additional features.

Here we propose a novel method called GraphProt2 that uses a graph convolutional neural network (GCN). GraphProt2 encodes input sequences as graphs, allowing the addition of base pair information in the form of graph edges. Furthermore, it supports variable size input and position-wise features like unpaired probabilities, conservation scores, or region type information. Compared to GraphProt, GraphProt2 offers an improved profile prediction mode, i.e., the calculation of position-wise prediction scores over whole RNA sequences, which has proven to be of great practical value in studies such as [23, 24], and to our knowledge no other tool offers. In the following we demonstrate its superior performance, comparing to GraphProt as well as the CNN-based method iDeepS, on a set of single CLIP-seq datasets and a combined dataset to learn a generic model.

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

Methods

Method overview

GraphProt2 utilizes RBP binding sites identified by CLIP-seq or similar methods to train a graph convolutional neural network (GCN) based model which can later be used to predict new binding sites on given input RNA sequences. Table 1 depicts some key attributes of GraphProt2 compared to GraphProt and current CNN-based methods. Unlike CNN methods, GraphProt2 offers both whole site and profile prediction for an input sequence, i.e., to predict one score for the whole sequence or individual scores for each nucleotide position of the sequence. Moreover, by using a GCN, base pair information can be included and input sequences can be of variable length, which makes the method more flexible and also enables the use of variable size windows in profile prediction. Finally, as with CNN-based methods, it supports additional nucleotide-wise features such as evolutionary conservation scores or region type information to increase its predictive performance. Compared to GraphProt, GraphProt2 offers an advanced profile prediction implementation, utilizing several window sizes to incorporate both local and context sequence information into the positional scoring.

Figure 1 sketches the GraphProt2 model architecture. Given an input graph derived from a binding site sequence, representations of the graph are learned by the GCN via several graph convolution layers. This is followed by a multi layer perceptron (MLP) part comprised of fully connected layers. In the end the network outputs a score for each class, which reflects the likelihood of the site belonging to the class. These scores can then be used to decide for its class label given a threshold. In case additional nucleotide-wise features are given, the values for each nucleotide are stored in a node attribute vector and assigned to the corresponding input graph node. In the following, we formally describe graph neural networks (GNNs) and provide the definitions necessary to understand the applied graph convolution operations.

Neural network for graphs

Notation and definitions We denote matrices, vectors, and variables with bold uppercase, uppercase, and lowercase letters, respectively. We consider a graph as a tuple $G = (\mathbb{V}, \mathbb{E}, \mathbf{X})$, where $\mathbb{V}, \mathbb{E}$ are the sets of nodes and edges. $\mathbf{X} \in \mathbb{R}^{n \times d}$ is a node attribute matrix, where each row $\mathbf{X}_i$ is a real-valued vector of size d associated to node v_i of the graph. We define the adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ as $a_{ij} = 1 \iff (i, j) \in \mathbb{E}$, and 0 otherwise. $\mathbf{D}$ is the degree matrix, where $d_{ij} = \sum_j a_{ij}$ if $i = j$, and 0 otherwise.

The first concepts of GNNs were described in [25, 26]. Based on these concepts, many methods have been proposed later on [27–29]. For an overview of existing GNN methods, we recommend the following recent survey [30]. GraphProt2 employs a GCN, which is a special kind of GNN that uses graph convolutions. A general GCN architecture includes the three main components: graph convolutions, a readout phase, and fully connected layers. In the following, we briefly describe the first two components.

Graph convolutions A graph convolution has its architecture defined following the graph topology, in which nodes are considered as neurons and edges as links in the network. Each node is assigned a state and the graph convolutions aim at iteratively updating each node state over time. Different policies used to update node states define distinct graph convolutions [27, 28, 31]. Generally, graph convolutions employ the current state of a node together with the accumulation over its neighbouring states, within a pre-defined number of hops, to update the node state.

Given a graph G with $\mathbf{A}$, $\mathbf{X}$ as its adjacency and attribute matrix, we use the following graph convolution as described in [28]:

$$\mathbf{H}^{t+1} = f(\tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{H}^t \mathbf{W}^t),$$

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

where $\mathbf{H^0} = \mathbf{X}$ and L is the number of convolution layers with $t = 0 \ldots (L-1)$. $\mathbf{H}^t$ is the state matrix or convolution output at time t, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ with $\mathbf{I}$ being the identity matrix, and $\tilde{\mathbf{D}}$ is the degree matrix corresponding to $\tilde{\mathbf{A}}$. $\mathbf{W}^t$ is the weight matrix containing the trainable convolution parameters at time t, and f is the element-wise non-linear activation function.

Readout phase Following the graph convolutions is a readout phase, in which the variable-size convolution outputs are converted into fixed-size inputs for the fully connected layers. In particular, graph node representations are taken from all convolution outputs of a graph and converted into a vector of fixed size, consistent over all graphs. A number of readout methods have been proposed in [27, 32].

Tool benchmark sets construction

To construct the benchmark sets we extracted eCLIP data out of two cell lines (HepG2, K562) from the ENCODE project website [33] (November 2018 release). We directly used the genomic binding regions (genome assembly GRCh38) identified by ENCODE's in-house peak caller CLIPper, which are available in BED format for each RBP and each replicate, often for both cell lines (thus 4 replicate BED files per RBP). For the single model benchmark set, binding sites were further filtered by their log2 fold change (FC) to obtain $\sim$ 6,000 to 10,000 binding regions for each replicate. We next removed sites with length $>$ 0.75 percentile length and selected for each RBP the replicate set that contained the most regions, centered the sites, and extended them to make all sites equal length. We chose a binding site length of 101 nt (50 nt extension up- and downstream of center position) and randomly selected 30 sets.

For the generic model benchmark set, we first merged both replicates of each RBP cell type combination, keeping only the sites with the highest FC in case of overlapping sites. After filtering (FC = 1), centering, and extending sites to 61 nt, we clustered the RBP cell type combinations (120 for K562, 104 for HepG2) by their binding site 3-mer content. We selected k=6 (maximum silhouette score), and selected 2 to 5 sets from each cluster, resulting in 20 datasets. After filtering (FC = 3) and randomly choosing 2,000 sites for each set, we again only kept the top FC sites in case of overlaps, and normalized the site lengths to 101 nt. This resulted in a set of 38,978 RBP binding sites from 20 different RBPs.

To generate the negative sets, we randomly selected sites based on two criteria: 1) their location on genes covered by eCLIP peak regions and 2) no overlap with any eCLIP peak regions from the experiment. The same number of random negative and positive instances was used throughout the benchmarks.

Tool setup for benchmarking

Both GraphProt2 and iDeepS were benchmarked using their default parameters. For GraphProt, hyperparameters were optimized for each model before cross validation, using a separate optimization set of 500 positive and 500 negative examples. Since iDeepS has no inherent cross validation mode, we manually split the data and ran the program 10 times on the respective train and test split sets.

Secondary structure information

GraphProt2 can include two kinds of structure information for a given RNA sequence: 1) base pairs and 2) unpaired probabilities for different loop contexts (probabilities for the nucleotide being paired or inside external, hairpin, internal or multi loops). Both are calculated using the ViennaRNA Python 3 API (ViennaRNA 2.4.13) and RNAplfold with its sliding window approach, with user-definable parameters (by default

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

these are window size = 150, maximum base pair span length = 100, and probabilities for regions of length u = 3). The base pairs with a probability ≥ a set threshold (default = 0.5) are then added to the sequence graph as edges, and the unpaired probability values are added to the node feature vectors.

Additional nucleotide-wise features

GraphProt2 supports both evolutionary conservation scores (phastCons and phyloP) and transcript region type information (one-hot encoded exon and intron labels) as nucleotide-wise features which are added to the node feature vectors. Conservation scores were downloaded from the UCSC website, using the phastCons and phyloP scores generated from multiple sequence alignments of 99 vertebrate genomes to the human genome (hg38, phastCons100way and phyloP100way datasets). Transcript region type labels (exon or intron) were assigned to each binding site position by taking a set of genomic exon regions and overlapping it with the genomic binding sites using intersectBed (bedtools 2.25.0). To unambiguously assign region type labels, we used the most prominent isoform for each gene. We defined the most prominent isoform of a gene based on the information Ensembl (Ensembl Genes 97, GRCh38.p12) provides for each transcript through hierarchical filtering: given that the transcript is part of the GENCODE basic gene set, we filtered by APPRIS annotation [34] (highest priority), transcript support level, and finally by transcript length (longer isoform preferred). The selected isoform exons were then used for region type assignment. Both conservation and region type features can thus be extracted by GraphProt2 for any given genomic binding site. In the case of spliced transcripts, their genomic exon regions can be supplied and GraphProt2 can reconstruct the full length RNA for prediction.

Measuring model performances

The accuracy measure, i.e., the proportion of correctly classified instances, was used in combination with 10-fold cross validation to estimate and compare single model generalization performances. Standard deviation of the 10 measured accuracies is reported for each model together with the average accuracy in the case of GraphProt2 and iDeepS. Since GraphProt offers a cross validation mode but does not output single fold accuracies, we only report its average accuracies. For generic models with n RBPs, the sites of each RBP were selected once for testing, while the remaining $n-1$ RBP sites were used for training, in total generating n different accuracy measures.

Tool requirements

GraphProt2 is implemented in Python 3. It uses the PyTorch framework [35] in combination with its extension library PyTorch Geometric [36], which supports deep learning for graphs. It is recommended to run GraphProt2 using an NVIDIA GPU (CUDA 10 support required) to speed up computations, although running it completely on CPU is also possible. GraphProt2 will soon be freely available on github, together with complete installation and usage instructions.

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

Results and Discussion

In the following, we demonstrate GraphProt2's superior performance based on two benchmarks, one over 30 single eCLIP RBP datasets, and the other over a combined set containing data from 20 RBPs. We compared GraphProt2 with GraphProt, a graph kernel approach that uses an SVM classifier, and iDeepS [37], a CNN-based method that also incorporates a long short-term memory (LSTM) architecture. For iDeepS, we trained models using both sequence and structure information. For GraphProt, we know from former studies that sequence models usually perform similar to structure models, while taking considerably less time for training. We therefore chose to train sequence models for the comparison. For GraphProt2, we found that incorporating structure into the graphs also did not significantly change the performance for the described benchmark sets. We therefore restricted the node features to nucleotide labels, conservation scores, and region type information for the two benchmarks.

GraphProt2 performs superior over single models

Table 2 lists the 10-fold cross validation results over 30 single eCLIP sets for GraphProt, iDeepS, and GraphProt2. GraphProt2 achieves the highest total average accuracy (86.43% ± 0.81), followed by iDeeps (82.24% ± 1.21), and GraphProt (74.66%, no single fold accuracies as described). This substantial increase in accuracy clearly demonstrates the power of deep learning methods compared to earlier state-of-the-art methods like GraphProt. In all 30 cases, GraphProt is outperformed by the deep learning methods. GraphProt2 shows better performance in 17 cases, while 9 cases are ties and in 5 cases iDeepS performs best.

Looking closer at the 17 cases, we can often see drastic improvements ($> 10\%$), for example in the case of DDX55, IGF2BP1, LIN28B, or UPF1. Given that all three methods use the same sequences for training, we found that these large performance increases can be mainly attributed to either one or two of the additional feature types (conservation or region type information). However, we cannot observe a general trend, which means that these features do not just capture common biases between positive and negative sets but are, depending on the RBP, indeed informative. As for the 5 cases in which iDeepS performs best, the performance increase is less pronounced, with a maximum of $\sim 4\%$. Besides architectural differences and the added structure information, the increase could also be due to the incorporated LSTM, which in theory should be able to better identify recurring patterns, but in practice we could not measure its contribution as it cannot be disabled.

It is known that RNA structure can be important for RBP binding [38, 39]. Since structure features did not significantly improve performance in GraphProt2, it could be argued that deep learning methods are powerful enough to detect discriminative structure information directly from the sequence data, whereas in earlier methods like GraphProt, predicted structures were still shown to boost performance for a small number of RBP datasets. Other reasons for the ineffectiveness of structure features could be that they are after all computationally predicted, and that CLIP-seq protocols tend to recover less structured binding sites because crosslinking of double-stranded regions is less efficient. Also, RNA structure itself is highly dynamic and affected by a multitude of interacting RBPs in the cell, which is currently not modelled by any prediction method. In addition, it is not clear whether the structure encoding chosen for a model is optimal for the task. Approaches like adding experimental structure probing data to support predictions, or CLIP-seq protocols that better capture structured binding sites might be a way to reevaluate the importance of structure in RBP binding site prediction. For example, hiCLIP [40] can identify double-stranded regions bound by an RBP, and irCLIP [41] can potentially resolve RNA secondary structures to increase

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

crosslinking efficiency for structured sites. However, these protocols have not yet been widely applied.

GraphProt2 generic model outperforms other methods

Table 3 presents the generic model results over a combined eCLIP set, consisting of sites from 20 different RBPs. As with the single models, GraphProt2 obtains the highest average accuracy (82.49%), followed by iDeepS (74.19%), and GraphProt (72.17%). Out of the 20 test comparisons, GraphProt2 achieves the highest accuracy in 13 cases. Furthermore, there are 6 tie cases between GraphProt2 and the other methods and one case where iDeepS performs best. Looking at the benchmark results, we can see that GraphProt2 increases its average accuracy lead from $\sim$ 4% (in Table 2) to $\sim$ 8%, while GraphProt is closer to iDeepS.

As described, the 20 RBPs were chosen based on k-means clustering of the 3-mer contents of their binding site sequences, in order to create a training set that contains a diverse collection of binding sites from RBPs with different binding preferences. This way we wanted to mitigate biases that would be introduced by random sampling of RBPs, assuming that many RBPs share similar binding preferences. Indeed, we observe that not all RBPs work well as test sets. There are particularly weak performing RBP sets over all three methods, such as for KHDRBS1, QKI, or HNRNPA1. Since we do not experience these drops with their single models, we can assume that these test sets indeed contain useful information, although the information does not seem to be common to most other RBPs in the training set. These varying performances also speak against a strong protocol-specific bias inherent to eCLIP data, which, if present, should result in more similar performances.

Apart from KHDRBS1, GraphProt2 often performs notably better on sets that show low performances ($\sim$ 70% or less) for iDeepS and GraphProt (DDX55, IGF2BP1, LIN28B, TIA1, and U2AF2). As with the single models, this effect can be attributed to the added conservation and region type information. One could argue that RBP binding sites are naturally biased towards conserved regions or specific region types, which leads to better accuracy scores. On the other hand, these biases also display generic properties of RBP binding sites, and thus are indeed valid features to use. This is especially true when the goal is to train a generic model, which needs to discriminate between common RBP binding and non-binding sites.

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

Conclusion

In this work we presented GraphProt2, a versatile deep learning-based RBP binding site prediction method which supports variable length input and additional nucleotide-wise features to achieve state-of-the-art predictive performance. Compared to popular CNN methods, the ability to work with variable length sequences makes GraphProt2 both more flexible and more accurate. Base pair annotation as well as nucleotide-wise loop context probabilities are also supported, although our results did not show any performance improvements for the constructed eCLIP benchmark datasets when adding these features. Taken together, the comprehensive feature and profile prediction support makes GraphProt2 a flexible and practical RBP binding site prediction tool ready to be utilized in future studies on RBP binding.

Tables

Table 1. Key attributes of GraphProt2 compared to GraphProt and current CNN-based methods. Note that compared to GraphProt, GraphProt2 offers a more sophisticated profile prediction implementation.

Attribute	GraphProt2	GraphProt	CNN-based methods
Model architecture	GCN	Graph kernel + SVM	CNN
Additional nucleotide features	YES	NO	YES
Built-in profile prediction	YES	YES	NO
Variable length input	YES	YES	NO
Base pair annotation	YES	YES	NO

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

Table 2. Single model benchmark results over 30 individual RBP eCLIP sets for GraphProt, iDeepS, and GraphProt2. We report average accuracies obtained by 10-fold cross validation together with standard deviations (apart from GraphProt).

RBP	GraphProt	iDeepS	GraphProt2
AGGF1	70.29	78.65±1.56	**84.29±1.10**
BUD13	70.06	79.78±0.87	**88.66±0.62**
CSTF2T	83.58	**89.89±0.60**	**89.62±0.46**
DDX55	67.57	72.58±1.44	**85.50±0.69**
EFTUD2	78.52	82.58±1.27	**85.44±1.01**
EWSR1	75.65	**82.83±0.79**	**82.98±1.08**
FASTKD2	71.26	79.84±1.35	**90.31±0.53**
FMR1	74.46	82.42±1.65	**92.62±0.77**
FUS	71.51	**78.55±1.59**	**78.39±0.73**
FXR2	76.86	84.44±0.87	**96.36±0.53**
HNRNPA1	76.01	**85.90±1.07**	81.09±1.06
HNRNPC	82.29	**91.82±0.65**	89.32±0.99
HNRNPK	88.63	**93.76±0.54**	**93.10±0.59**
IGF2BP1	68.64	80.46±1.45	**92.97±0.74**
KHDRBS1	76.95	**82.64±1.61**	**82.03±0.75**
LIN28B	66.89	76.45±1.08	**90.13±0.66**
PCBP2	86.94	**93.17±0.78**	**93.16±0.42**
PTBP1	84.27	**89.40±0.85**	**90.04±1.01**
PUM2	58.06	65.79±2.02	**70.55±1.13**
QKI	75.86	**84.32±1.06**	**83.88±1.10**
RBFOX2	67.80	76.12±1.10	**78.44±1.24**
SF3B4	70.46	78.78±0.76	**90.19±0.56**
SFPQ	67.31	70.63±2.74	**74.47±1.58**
SMNDC1	77.86	81.37±2.01	**83.56±0.51**
SRSF1	82.64	90.06±0.45	**92.08±1.10**
TAF15	78.24	**83.66±1.01**	**83.58±0.97**
TARDBP	91.39	**94.20±0.95**	92.91±0.74
TIA1	68.64	80.67±1.09	**82.16±0.74**
U2AF2	69.88	**86.88±0.76**	83.24±0.99
UPF1	61.47	69.70±2.33	**91.91±0.80**
AVG	74.66	82.24±1.21	**86.43±0.81**

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

Table 3. Generic model benchmark results over a combined eCLIP set containing sites from 20 different RBPs, for GraphProt, iDeepS, and GraphProt2. In each round a model was trained on 19 RBP sets and tested on the remaining RBP set. We report test accuracies for each round.

RBP	GraphProt	iDeepS	GraphProt2
AGGF1	73.09	72.83	**79.14**
CSTF2T	**88.67**	86.48	**89.62**
DDX55	72.93	72.50	**85.02**
EWSR1	**88.56**	**88.34**	**88.10**
FMR1	79.65	80.03	**89.77**
FUS	**86.87**	85.61	**87.75**
FXR2	80.16	79.75	**92.58**
HNRNPA1	57.92	64.08	**70.67**
HNRNPK	85.15	**88.59**	85.87
IGF2BP1	70.62	73.67	**92.53**
KHDRBS1	42.41	45.91	**53.28**
LIN28B	67.35	65.38	**88.66**
PCBP2	86.79	84.92	**89.22**
PTBP1	69.54	**77.61**	**78.82**
PUM1	76.85	74.52	**91.97**
QKI	49.87	59.23	**69.03**
TAF15	73.81	**74.99**	**75.75**
TARDBP	**75.26**	69.99	**76.10**
TIA1	60.68	67.78	**83.24**
U2AF2	57.24	71.57	**82.61**
AVG	72.17	74.19	82.49

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

Figures

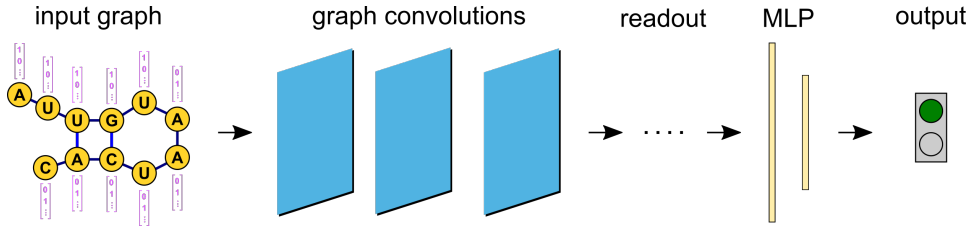

Figure 1. GraphProt2 model architecture schematic.

Acknowledgments

We thank Daniel Maticzka for his initial work on the method and Martin Raden for his invaluable suggestions on the topic.

Funding

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2189 – Project ID: 390939984, BA 2168/11-1 SPP 1738 and BA2168/11-2 SPP 1738, BA 2168/3-3, and GRK 2344/1 2017 MeInBio.

Diese Arbeit wurde gefördert durch die Deutsche Forschungsgemeinschaft (DFG) im Rahmen der Exzellenzstrategie des Bundes und der Länder – EXC-2189 – Projektnummer 390939984.

References

1. Stefanie Gerstberger, Markus Hafner, and Thomas Tuschl. A census of human rna-binding proteins. *Nature Reviews Genetics*, 15(12):829, 2014.

2. Kristopher W Brannan, Wenhao Jin, Stephanie C Huelga, Charles AS Banks, Joshua M Gilmore, Laurence Florens, Michael P Washburn, Eric L Van Nostrand, Gabriel A Pratt, Marie K Schwinn, et al. Sonar discovers rna-binding proteins from analysis of large-scale protein-protein interactomes. *Molecular cell*, 64(2):282–293, 2016.

3. Matthias W Hentze, Alfredo Castello, Thomas Schwarzl, and Thomas Preiss. A brave new world of rna-binding proteins. *Nature Reviews Molecular Cell Biology*, 19(5):327, 2018.

4. Lichao Liu, Tong Li, Guang Song, Qingxia He, Yafei Yin, J Yuyang Lu, Xianju Bi, Kaili Wang, Sai Luo, Yu-Sheng Chen, et al. Insight into novel rna-binding activities via large-scale analysis of lncrna-bound proteome and idh1-bound transcriptome. *Nucleic acids research*, 47(5):2244–2262, 2019.

5. Stefanie Gerstberger, Markus Hafner, Manuel Ascano, and Thomas Tuschl. Evolutionary conservation and expression of human rna-binding proteins and

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

their role in human genetic disease. In *Systems biology of RNA binding proteins*, pages 1–55. Springer, 2014.

6. Bruno Pereira, Marc Billaud, and Raquel Almeida. Rna-binding proteins in cancer: old players and new actors. *Trends in cancer*, 3(7):506–528, 2017.

7. Erin G Conlon and James L Manley. Rna-binding proteins in neurodegeneration: mechanisms in aggregate. *Genes & development*, 31(15):1509–1528, 2017.

8. Donny D Licatalosi, Aldo Mele, John J Fak, Jernej Ule, Melis Kayikci, Sung Wook Chi, Tyson A Clark, Anthony C Schweitzer, John E Blume, Xuning Wang, et al. Hits-clip yields genome-wide insights into brain alternative rna processing. *Nature*, 456(7221):464, 2008.

9. Markus Hafner, Markus Landthaler, Lukas Burger, Mohsen Khorshid, Jean Hausser, Philipp Berninger, Andrea Rothballer, Manuel Ascano Jr, Anna-Carina Jungkamp, Mathias Munschauer, et al. Transcriptome-wide identification of rna-binding protein and microrna target sites by par-clip. *Cell*, 141(1):129–141, 2010.

10. Julian König, Kathi Zarnack, Gregor Rot, Tomaž Curk, Melis Kayikci, Blaž Zupan, Daniel J Turner, Nicholas M Luscombe, and Jernej Ule. iclip reveals the function of hnrnp particles in splicing at individual nucleotide resolution. *Nature structural & molecular biology*, 17(7):909, 2010.

11. Eric L Van Nostrand, Gabriel A Pratt, Alexander A Shishkin, Chelsea Gelboin-Burkhart, Mark Y Fang, Balaji Sundararaman, Steven M Blue, Thai B Nguyen, Christine Surka, Keri Elkins, et al. Robust transcriptome-wide discovery of rna-binding protein binding sites with enhanced clip (eclip). *Nature methods*, 13(6):508, 2016.

12. Michael Uhl, Torsten Houwaart, Gianluca Corrado, Patrick R Wright, and Rolf Backofen. Computational analysis of clip-seq data. *Methods*, 118:60–72, 2017.

13. Philip J Uren, Emad Bahrami-Samani, Suzanne C Burns, Mei Qiao, Fedor V Karginov, Emily Hodges, Gregory J Hannon, Jeremy R Sanford, Luiz OF Penalva, and Andrew D Smith. Site identification in high-throughput rna–protein interaction data. *Bioinformatics*, 28(23):3013–3020, 2012.

14. Michael T Lovci, Dana Ghanem, Henry Marr, Justin Arnold, Sherry Gee, Marilyn Parra, Tiffany Y Liang, Thomas J Stark, Lauren T Gehman, Shawn Hoon, et al. Rbfox proteins regulate alternative mRNA splicing through evolutionarily conserved RNA bridges. *Nature structural & molecular biology*, 20:1434, 2013.

15. Thorsten Bischler, Daniel Maticzka, Konrad U. Förstner, and Patrick R. Wright. PEAKachu. `https://github.com/tbischler/PEAKachu`.

16. Sabrina Krakau, Hugues Richard, and Annalisa Marsico. PureCLIP: capturing target-specific protein–RNA interaction footprints from single-nucleotide CLIP-seq data. *Genome biology*, 18(1):240, 2017.

17. Aleksandra E Kornienko, Christoph P Dotter, Philipp M Guenzl, Heinz Gisslinger, Bettina Gisslinger, Ciara Cleary, Robert Kralovics, Florian M Pauler, and Denise P Barlow. Long non-coding rnas display higher natural expression variation than protein-coding genes in healthy humans. *Genome biology*, 17(1):14, 2016.

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

18. Roberto Ferrarese, Griffith R Harsh, Ajay K Yadav, Eva Bug, Daniel Maticzka, Wilfried Reichardt, Stephen M Dombrowski, Tyler E Miller, Anie P Masilamani, Fangping Dai, et al. Lineage-specific splicing of a brain-enriched alternative exon promotes glioblastoma progression. *The Journal of clinical investigation*, 124(7):2861–2876, 2014.

19. Hilal Kazan, Debashish Ray, Esther T Chan, Timothy R Hughes, and Quaid Morris. Rnacontext: a new method for learning the sequence and structure binding preferences of rna-binding proteins. *PLoS computational biology*, 6(7):e1000832, 2010.

20. Daniel Maticzka, Sita J Lange, Fabrizio Costa, and Rolf Backofen. Graphprot: modeling binding preferences of rna-binding proteins. *Genome biology*, 15(1):R17, 2014.

21. Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. Predicting the sequence specificities of dna-and rna-binding proteins by deep learning. *Nature biotechnology*, 33(8):831, 2015.

22. Xiaoyong Pan, Yang Yang, Chun-Qiu Xia, Aashiq H Mirza, and Hong-Bin Shen. Recent methodology progress of deep learning for rna–protein interaction prediction. *Wiley Interdisciplinary Reviews: RNA*, page e1544, 2019.

23. Yashar S Niknafs, Sumin Han, Teng Ma, Corey Speers, Chao Zhang, Kari Wilder-Romans, Matthew K Iyer, Sethuramasundaram Pitchiaya, Rohit Malik, Yasuyuki Hosono, et al. The lncrna landscape of breast cancer reveals a role for dscam-as1 in breast cancer progression. *Nature communications*, 7:12791, 2016.

24. Alexander R Gawronski, Michael Uhl, Yajia Zhang, Yen-Yi Lin, Yashar S Niknafs, Varune R Ramnarine, Rohit Malik, Felix Feng, Arul M Chinnaiyan, Colin C Collins, et al. Mechrna: prediction of lncrna mechanisms from rna–rna and rna–protein interactions. *Bioinformatics*, 34(18):3101–3110, 2018.

25. Alessandro Sperduti and Antonina Starita. Supervised neural networks for the classification of structures. *IEEE Transactions on Neural Networks*, 8(3):714–735, 1997.

26. Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

27. Muhan Zhang, Zhicheng Cui, Marion Neumann, and Yixin Chen. An end-to-end deep learning architecture for graph classification. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

28. Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

29. Nicolò Navarin, Dinh V Tran, and Alessandro Sperduti. Pre-training graph neural networks with kernels. *arXiv preprint arXiv:1811.06930*, 2018.

30. Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S Yu. A comprehensive survey on graph neural networks. *arXiv preprint arXiv:1901.00596*, 2019.

31. Dinh V Tran, Nicolò Navarin, and Alessandro Sperduti. On filter size in graph convolutional networks. In *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1534–1541. IEEE, 2018.

bioRxiv preprint doi: https://doi.org/10.1101/850024; this version posted November 21, 2019. The copyright holder for this preprint (which was not certified by peer review) is the author/funder, who has granted bioRxiv a license to display the preprint in perpetuity. It is made available under aCC-BY-NC-ND 4.0 International license.

32. Nicolò Navarin, Dinh Van Tran, and Alessandro Sperduti. Universal readout for graph convolutional neural networks. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2019.

33. Cricket A Sloan, Esther T Chan, Jean M Davidson, Venkat S Malladi, J Seth Strattan, Benjamin C Hitz, Idan Gabdank, Aditi K Narayanan, Marcus Ho, Brian T Lee, et al. Encode data at the encode portal. *Nucleic acids research*, 44(D1):D726–D732, 2015.

34. Jose Manuel Rodriguez, Paolo Maietta, Iakes Ezkurdia, Alessandro Pietrelli, Jan-Jaap Wesselink, Gonzalo Lopez, Alfonso Valencia, and Michael L Tress. Appris: annotation of principal and alternative splice isoforms. *Nucleic acids research*, 41(D1):D110–D117, 2012.

35. Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

36. Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

37. Xiaoyong Pan, Peter Rijnbeek, Junchi Yan, and Hong-Bin Shen. Prediction of rna-protein sequence and structure binding preferences using deep convolutional and recurrent neural networks. *BMC genomics*, 19(1):511, 2018.

38. Eckhard Jankowsky and Michael E Harris. Specificity and nonspecificity in rna–protein interactions. *Nature reviews Molecular cell biology*, 16(9):533–544, 2015.

39. J Matthew Taliaferro, Nicole J Lambert, Peter H Sudmant, Daniel Dominguez, Jason J Merkin, Maria S Alexis, Cassandra A Bazile, and Christopher B Burge. Rna sequence context effects measured in vitro predict in vivo protein binding and regulation. *Molecular cell*, 64(2):294–306, 2016.

40. Yoichiro Sugimoto, Alessandra Vigilante, Elodie Darbo, Alexandra Zirra, Cristina Militti, Andrea D'Ambrogio, Nicholas M Luscombe, and Jernej Ule. hiclip reveals the in vivo atlas of mrna secondary structures recognized by staufen 1. *Nature*, 519(7544):491, 2015.

41. Brian J Zarnegar, Ryan A Flynn, Ying Shen, Brian T Do, Howard Y Chang, and Paul A Khavari. irclip platform for efficient characterization of protein–rna interactions. *Nature methods*, 13(6):489, 2016.