LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT
TECHNISCHE UNIVERSITÄT MÜNCHEN TUM

Technische Universität München

Lehrstuhl für Bioinformatik, I12

Diplomarbeit

in Bioinformatik

Improvement of DNA- and RNA-Protein Binding Prediction

Peter Hönigschmid

Aufgabensteller: Prof. Dr. Burkhard Rost

Betreuer: Dr. Edda Kloppmann

Abgabedatum: 16.08.2012

Ich versichere, dass ich diese Diplomarbeit selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

16.08.2012 _____

Peter Hönigschmid

Abstract: Polynucleotide-protein interactions play an important role in many essential molecular processes, especially those dealing with the synthesis of proteins. A polynucleotide is either deoxyribonucleic acid (DNA) or ribonucleic acid (RNA). Transcription factors are a prominent example for proteins binding directly to DNA to regulate gene expression. As is known today, there are manifold post-transcriptional modifications made to the RNA such as alternative splicing, which are initiated by RNA-binding proteins, like the spliceosome, a eukaryotic protein-RNA complex. RNA-protein-complexes, e.g. ribosomes, are involved in a multitude of important processes in the cell.

Although there are already various methods to predict polynucleotide binding on a per-residue basis showing good performance, none of them deals with the differentiation of proteins that bind polynucleotides, and those which do not. The approach presented in this work handles this problem by using neural networks together with a clean dataset containing proteins that are definitively not involved in polynucleotide binding. This allows distinguishing between proteins that bind DNA, RNA or none of those combined with a residue-based prediction of the binding. These predictions shall serve the demand for experimentalists to find new targets involved in those essential molecular processes followed by an identification of polynucleotide binding regions inside those proteins in one tool.

Zusammenfassung: Interaktionen zwischen Polynukleotiden und Proteinen spielen eine wichtige Rolle in vielen essentiellen molekularen Vorgängen, im speziellen jene die an der Proteinbiosynthese beteiligt sind. Ein Polynukleotid ist entweder Desoxyribonukleinsäure (DNA) oder Ribonukleinsäure (RNA). Transkriptionsfaktoren sind ein bekanntest Beispiel für Proteine die DNA direkt binden um die Genexpression in jeglicher Richtung zu regulieren. Wie heute bekannt ist, gibt es vielfältige posttranskriptionale Modifikationen an der RNA, wie zum Beispiel alternatives Splicen, welche wiederum von RNA bindenden Proteinen initiiert werden, in diesem speziellen Fall dem eukaryotischen Spliceosom. RNA-Protein-Komplexe sind an einer Vielzahl bedeutender zellulärer Prozesse beteiligt.

Auch wenn bereits einige Vorhersagemethoden existieren, welche gute Genauigkeit bezüglich der reste-basierten Vorhersage liefern, ist keine davon in der Lage zwischen Proteinen zu unterscheiden die Polynukleotide binden, und solchen die es nicht tun. Der Ansatz den diese Arbeit verfolgt, befasst sich mit diesem Problem durch die Benutzung von neuronalen Netzwerken zusammen mit einem Datensatz das ebenfalls Proteine enthält die mit Sicherheit keinen Polynukleotide binden. Diese Vorhersagen sollen der Anforderung von experimentell arbeitenden Wissenschaftlern nachkommen, neue Proteine zu finden die an diesen essentiellen molekularen Vorgängen beteiligt sind, kombiniert mit der Bestimmung Polynukleotid-bindender Regionen, mit Hilfe ein und desselben Werkzeugs.

Acknowledgements: First of all, I would like to thank Burkhard Rost for making the thesis possible, his support by answering all my questions (if an answer existed) and for his help to raise my confidence in my skills for and comprehension of the bioinformatics field. Thanks to Edda Kloppmann for her time and advise regarding the writing of this thesis as well as giving me insights due to her profound scientific knowledge. I also want to thank Christian Schaefer and Maximilian Hecht for a lot of machine-learning related talk. They provided me with new ideas and helped saving time by avoiding mistakes even before I made them. Further gratitude is due to Tim Karl, Laszlo Kajan and Marlena Drabik for enabling a flawless everyday business. Additional thanks to the whole Rostlab for making me feel as a part of a great work and social environment.
Last but not least, many special thanks to my family for supporting me in any imaginable way during my studies and wherever I may roam in the future.

Introduction

Polynucleotides

The two polynucleotides desoxyribonucleic acid (DNA) and ribonucleic acid (RNA) are maybe the most important molecules which ever arose on earth: Due to the RNA world hypothesis, they were no less than the origin of life (Gilbert 1986) with RNA being the first molecule to replicate and to catalyze its own replication. The function of DNA and RNA today are no less important but much more versatile. DNA (and sometimes RNA) allows the transfer of genetic information from one generation to another in any species. During this transfer, it is often recombined between the individuals or changed by chance, i.e. mutated, and thus gives the opportunity for adaption to the environment, which changes permanently. Furthermore, today humans can use DNA for their benefit through genetic engineering, e.g. causing bacteria to produce insulin, an important drug to treat diabetes. RNA often acts as a messenger to realize/put into effect the information of the blueprint, namely the DNA. But the research of the last decades has also brought to attention, that RNA can do more than that. It is, for example, capable of transcription regulation, and even of catalyzing reactions itself.

Polynucleotide-Protein Binding

Although DNA and RNA are already powerful molecules, there is the possibility of interaction with proteins, leading to even more productive biological mechanisms. Several classes of proteins are involved in DNA or RNA binding (Bruce Alberts 2002):

Transcription factors are proteins that bind to a specific sequence of the DNA called promoter or enhancer region. Through this mechanism, the transcription factors regulate the expression of the gene, or at least its transcription. This regulation can either be positive, which makes the transcription factor an activator, or negative for a repressor. They are often dependent on initiation

factors consisting of RNA. An example of a transcription factor can be seen in Figure 1.

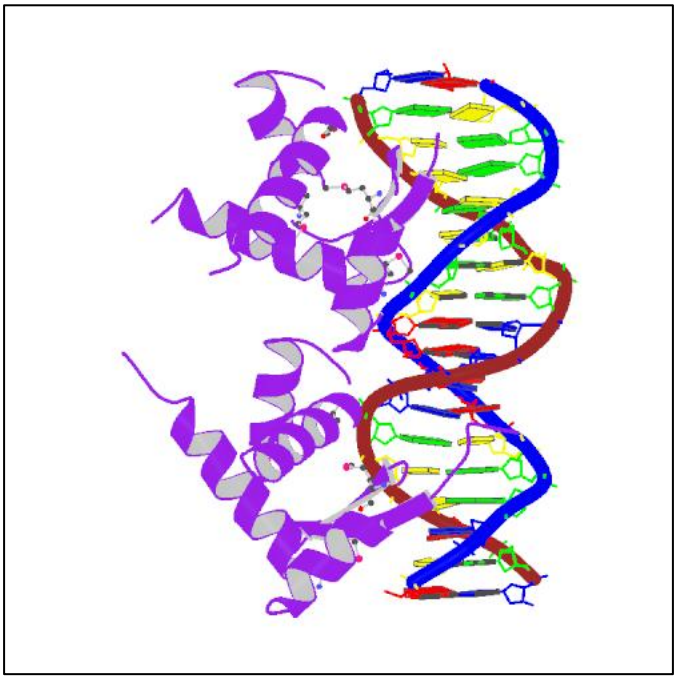


Figure 1: Crystal Structure of the Transcription Factor AmrZ in Complex with the 18 Base Pair amrZ1 Binding Site (E.E. Pryor 2012). This transcription factor functions both as an activator and repressor of multiple genes encoding Pseudomonas aeruginosa virulence factors.

Polymerases are enzymes participating in the replication of DNA and the transcription of RNA. Together with other proteins, they initiate the base-paring process building the complementary strand to a so-called template strand. This natural process is also used to amplify DNA in genetics. This process is called polymerase chain reaction (PCR) (RK Saiki 1985).

The enzyme **nuclease** can cleave DNA or RNA by splitting the phosphodiester bonds between the nucleic acid subunits. It is also specific to particular sequences (Daniel Nathans 1975). In genetic assays it is used to incorporate foreign polynucleotide fragments for example into plasmids.

Histones are proteins that are involved in DNA packaging in eukaryotic cells. The resulting structure is called nucleosome, which makes it possible for the

nucleus to store much more DNA than would be in its histone-free form (Figure 2).

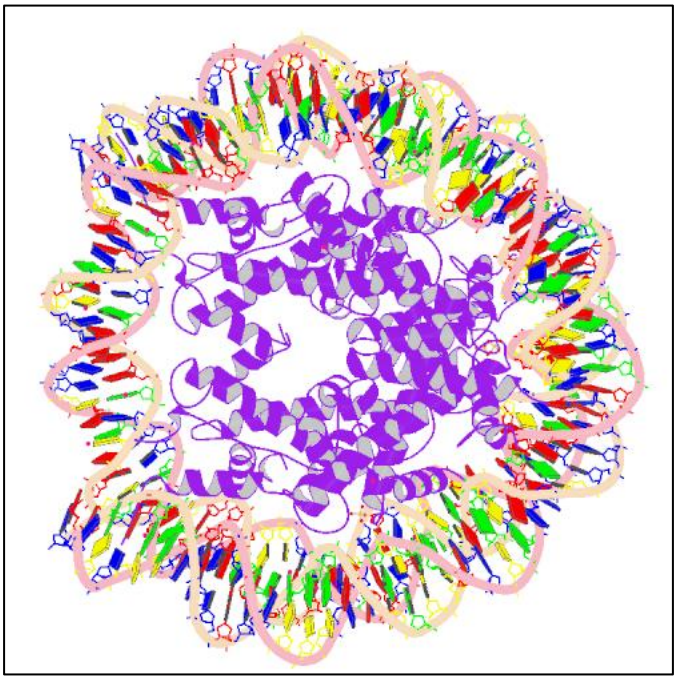


Figure 2: Crystal Structure of the Nucleosome Core Particle assembled with the 146b Alpha-Satellite Sequence (NCP146b) (E.Y.D. Chua 2012).

Splicing is a process taking place in eukaryotic cells after transcription and before translation. The molecule that is involved in this process is the spliceosome, a complex consisting of protein and RNA. During splicing events, the introns are cut out of the pre-mRNA to form the final mRNA, which is then translated to the protein.

Novelty Of The Method

SomeNA is the name of the method introduced in this thesis. This section covers the uniqueness of the tool compared to present polynucleotide-protein binding prediction tools.

There are three aspects that define such polynucleotide-protein binding prediction methods. The first differentiation takes place at the input the method uses. Binding can either be predicted from sequence alone like with SomeNA, or from given 3D coordinates like DR_bind (Yao Chi Chen 2012). Both approaches have their benefits, being either more precise according to their results (3D coordinates), or the possibility to apply the method to every sequenced protein available. SomeNA uses only sequence as input, which makes it more generally applicable.

The second aspect is the type of polynucleotide binding that is predicted, which can either be DNA, RNA or both. Recent methods, like RNABindR (R.R. Walia 2012) or DNABindR (Changhui Yan 2006), specialize in one of those while SomeNA has its strength in predicting the type of polynucleotide that is bound. Thirdly, the prediction methods can be separated into those that do residue-based prediction as DNABindR and RNABindR, and those that predict if a protein binds a polynucleotide in general. An example for such a protein-based prediction methods is DNAbinder (Manish Kumar 2007). Another novelty is the combination of residue- and protein-based prediction. While other tools focus on either one of them, SomeNA creates protein-based predictions out of residue-based ones by applying an unique scoring system that incorporates the natural occurrence of clusters of binding residues. Figure 3 illustrates the capabilities of the created method against the possible prediction aspects.

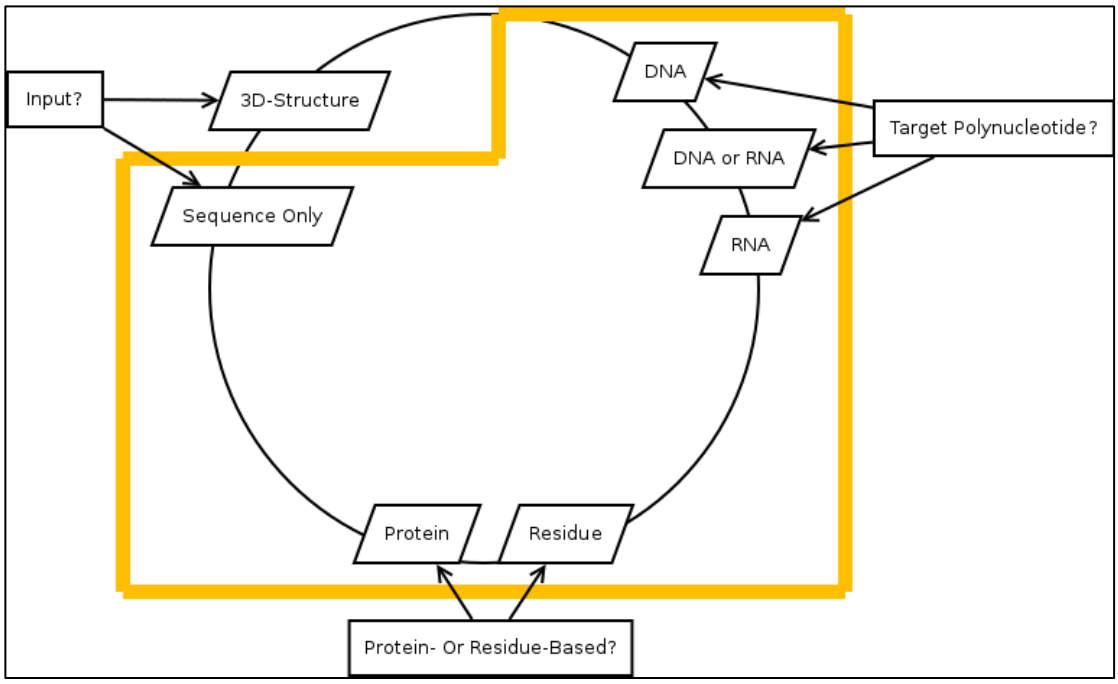


Figure 3: The different aspects of a polynucleotide-protein binding prediction method. Current methods cover only one choice of each aspect, e.g. protein- or residue-based DNA binding prediction, while the method introduced in this thesis can handle all sequence-based predictions, as is illustrated by the area enclosed in orange.

Methods and Computational Details

Artificial Neural Networks

An artificial neural network or neural network is a machine-learning method inspired by biological neural networks in terms of their functionality and structure. It is capable of learning relationships between given input data and the desired output or prediction. Expressed differently, it can approximate the function, which leads from the input data to the output. As in a real biological neural network, it consists of many neurons, which can also be referred to as **perceptrons**, units or nodes. Furthermore, the processing of signals has its analogies to the real world example, since a neuron receives many signals as input, and forwards a signal as long as the accumulated inputs are stronger than a certain threshold.

Basic Neural Network OR Linear Classification

In artificial neural networks, this principle of operation is nearly the same. The most basic neural network consists of just one perceptron, which receives several signals of different strength as input, sums those values up, and evaluates this sum using a threshold function, which calculates an output value for this perceptron. To make this perceptron capable of learning about the importance of the different input values, every input to this perceptron is weighted with a different multiplier. A schematic is depicted in Figure 4. For convenience, the input and output values will also be described as input/output units, neurons or nodes.

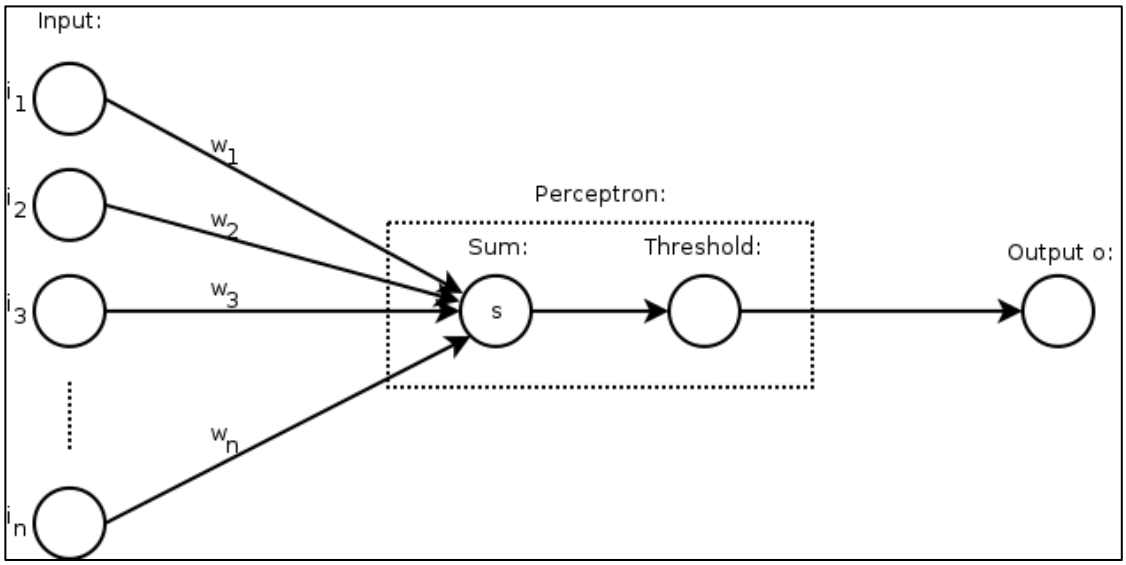


Figure 4: The schematic depicts a simple neural network with on perceptron receiving input signals i_n weighted according to their importance by w_n. The perceptron evaluates the sum of the signals using a threshold function. This threshold decides whether a signal is output.

The following formula describes the addition of the inputs i with the incorporation of the weights w, which are adjusted during the learning process.

$$s = \sum_{j=1}^{n} i_j * w_j$$

As threshold function the widely used sigmoid function as a special case of the logistic function is used. It is able to convert any incoming sum to a value between 0 and 1. In addition it provides finer granularity for the output than for example a simple step function, which gives either 0 or 1 as output.

$$o = \frac{1}{1 + e^{-s}}$$

This basic version of a neural network is able to solve linear classification tasks. Linear classification is characterized by the possibility to discriminate a number of points, where any point belongs to one of two classes, by inserting a hyperplane, which separates the two classes. A new unclassified data point, which has also its features values, can then be spotted either on the one or the other side of the hyperplane, and thus be classified as can be seen in Figure 5.

Non-Linear Classification

For non-linear classification like the task handled in this work more complex neural networks are employed. That is, if the problem becomes more difficult in terms of complexity, because of much bigger feature vectors and overlapping data points, the forming of a hyperplane is impossible. To deal with this problem, a non-linear classifier is used, which does not use a hyperplane anymore but a more complex polynomial.

Other machine-learning algorithms than neural networks, which are only able to do linear classification like support-vector machines (SVMs), try to handle this problem by extrapolating the feature vectors into a higher dimensional space using the kernel trick. The idea being that linear classification is possible in this higher dimensional feature space. However, the use of the kernel trick is also the caveat of SVMs and other linear classification algorithms, since calculating the final high dimensional feature space via the use of the kernel is time and computation intensive, which makes the training of the method as well as the classification itself fairly slow.

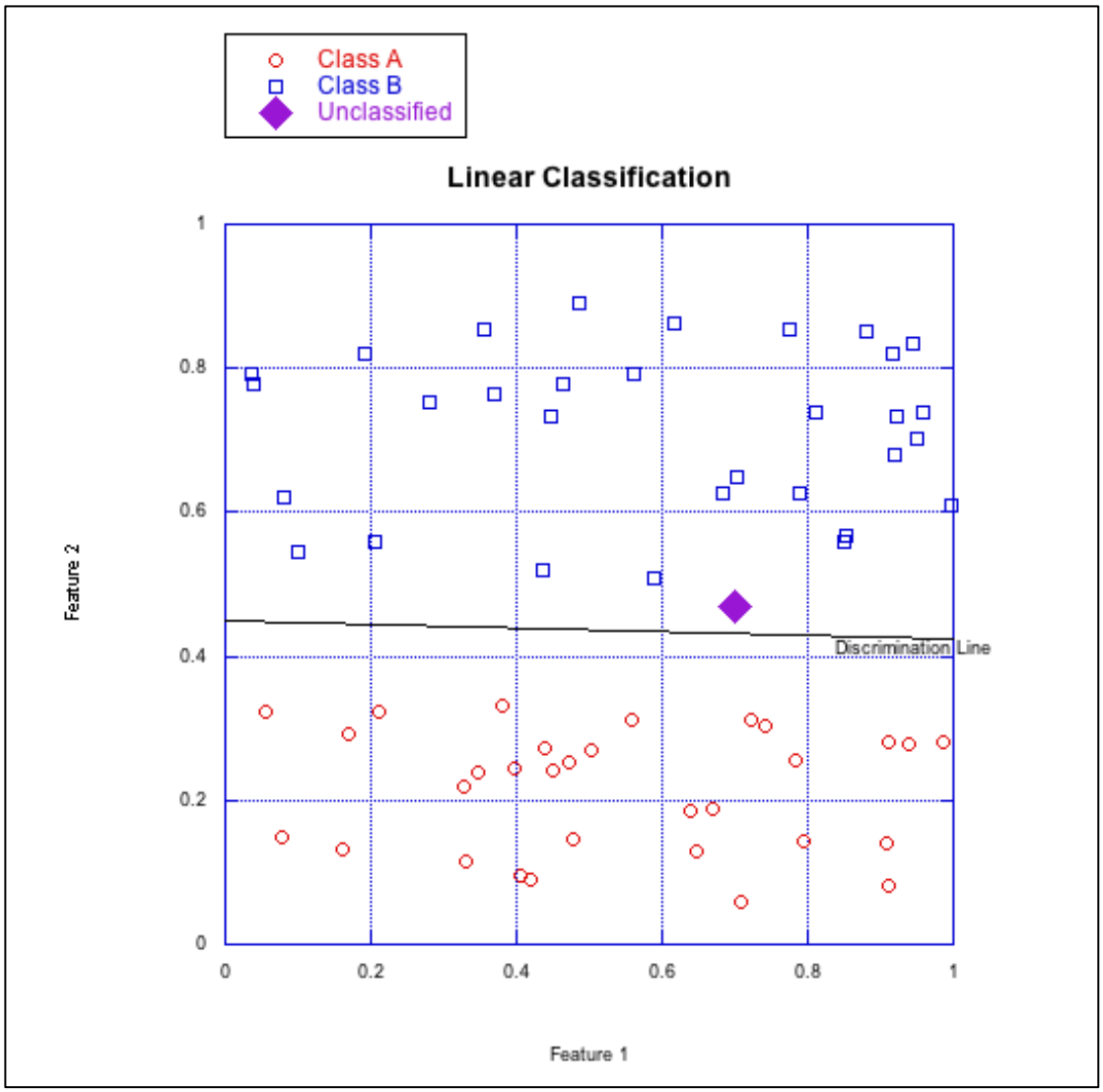


Figure 5: Using fictitious data, the plot shows two classes, A and B, in the feature space. The discrimination line indicates visually a possible separation of those two classes. By this line, the unclassified point can be assigned to class A.

Neural networks however do not need to extrapolate the feature vectors. To deal with a high-complexity problem, it is possible, and necessary, to create a network of perceptrons. In most applications, such a network consists of three so-called layers. The first layer, the input layer contains as many nodes as the length of the input feature vector is. This input layer provides these inputs to the hidden layer, no calculation is performed at this stage. The second layer, called hidden layer, consists of hidden neurons, where all of them are perceptrons. The number of perceptrons can vary between a few and many. There is a connection from every input neuron to every neuron of the hidden layer, each assigned its

own weight. The final layer, or output layer, again consists of perceptrons, present in the quantity of output classes, and again connected to each neuron of the hidden layer. This network topology is called a **fully connected feed forward network** (Figure 6).

Although in this work only the number of hidden units is changed in terms of the network topology, there is also the possibility to insert more hidden layers, to change the connection rate between the layers or to "feed back" the output of units to units in a preceding layer in the network.

In this work, a fully connected feed forward is applied to determine polynucleotide binding of proteins. As there are two possibilities for each residue in the protein, i.e. to bind a polynucleotide or not to bind, the network used in this work includes two output units (Figure 6).

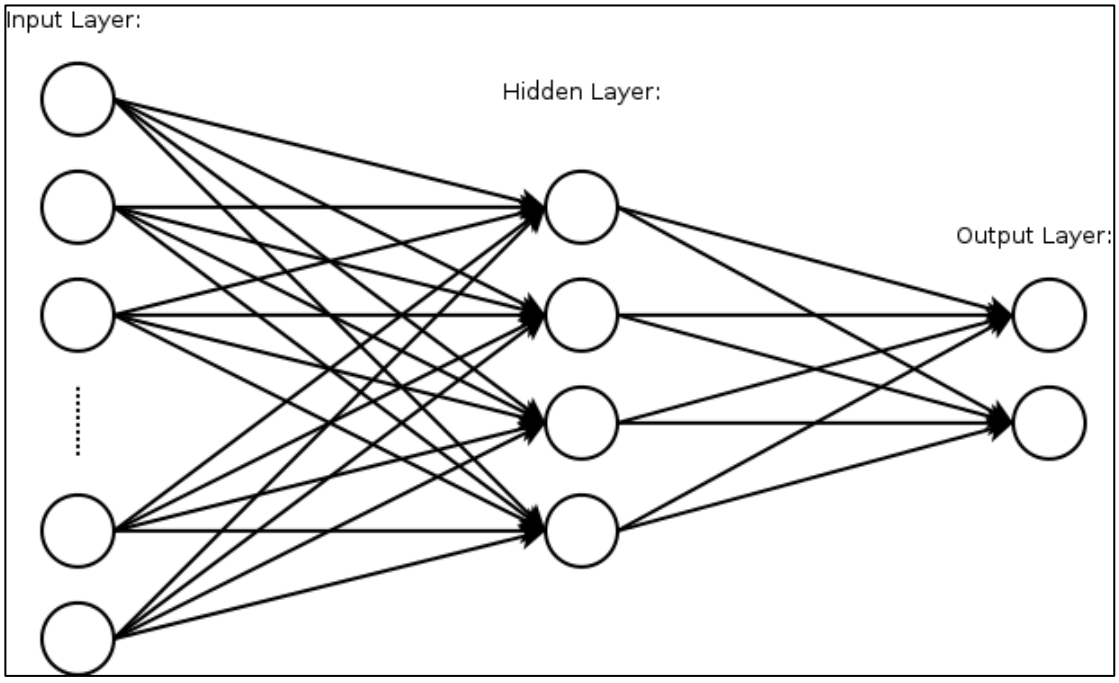


Figure 6: Fully connected feed forward network with the input layer, one hidden layer consisting of four units, and the output layer with two units.

Creating (OR Training) a Fully Connected Feed Forward Network

To create a neural network model suitable to the given data, it has to be trained. Apart from the network topology the training process describes mainly the adjustment of the weights, which are many.

$$|weights| = |input\ units| * |hidden\ units| * |output\ units|$$

The training of those weights is achieved by the on-line backward propagation of errors method, called **backpropagation** in short. There are two requirements for the algorithm to work. Firstly, the weights of the neural network have to be initialized with small random values. Secondly, the threshold function used in the perceptrons has to be differentiable. The algorithm relies on the principle of the stochastic gradient descent to find the right direction for adjusting the weights towards a minimum of the error surface landscape.

The backpropagation algorithm can be divided into two steps (Flowchart in Figure 7). The first step is called “propagation”, which includes the presentation of the training example to the network and propagating it up to the output nodes. This is the same procedure as applied for the final classification of a data point. The difference, or delta, of the expected output and the produced output is calculated and propagated backwards through the network to get the delta for all units.
The second step is the “weight update”, where the delta of all units is multiplied with the differentiation of the threshold function to get the gradient for this weight. This weight is then adjusted by using a factor called learning rate. The learning rate influences the speed at which the weights are adjusted. That is a high learning rate can lead to a fast convergence towards a local error minimum, with the disadvantage of being less accurate.

An additional parameter, the learning momentum, takes into account the weight changes of the previous learning iteration by increasing the speed of adaption of a weight into the same direction as in the iteration before.

```
Initialize the weights in the network
For each example e in the training set
    O = neural-net-output(network, e)
    T = teacher output for e
    Calculate error (T - O) at the output units
    Compute delta_wh for all weights from hidden layer to output
    Compute delta_wi for all weights from input layer to hidden layer
    Update the weights in the network
Until all examples classified correctly or stopping criterion satisfied
```

Figure 7: Simplified flowchart of the backpropagation algorithm. After initializing the weights in the network, iteratively several functions are applied to each example in the training set to update the weights until the stop criterion is satisfied.

For this work, the Fast Artificial Neural Network (Nissen 2012) C library is used coupled with the Perl language-binding module AI::FANN (García 2009), which is available via the Comprehensive Perl Archive Network (The Comprehensive Perl Archive Network 2012).

Datasets

To create a useful predictor for the underlying problem, the machine-learning algorithm depends on examples where the outcome is known. This is also necessary to test the performance of the prediction tool, making it comparable to methods already available. A well-curated dataset is therefore of major importance…

Interface Definition

The major step to create such datasets is the selection of suitable proteins, i.e. proteins, which are shown experimentally to bind polynucleotides and additionally a negative set, i.e. proteins, which are shown not to bind polynucleotides. Furthermore, the definition of an interacting residue is very important.

The basic approach is to measure the distances between every atom of the polynucleotide and those from the protein chain. If this distance is shorter than a specified threshold, the residue is declared as interacting residue. In addition, it is necessary to introduce another restriction to extract only so called "effective interactions" (E. Ferrada 2009), because it is possible that two interaction partners are indeed closer than the given threshold, but there are other atoms between them. In this case, it is more likely that those atoms are the interacting ones, because they shield the other interaction like shown in Figure 8.

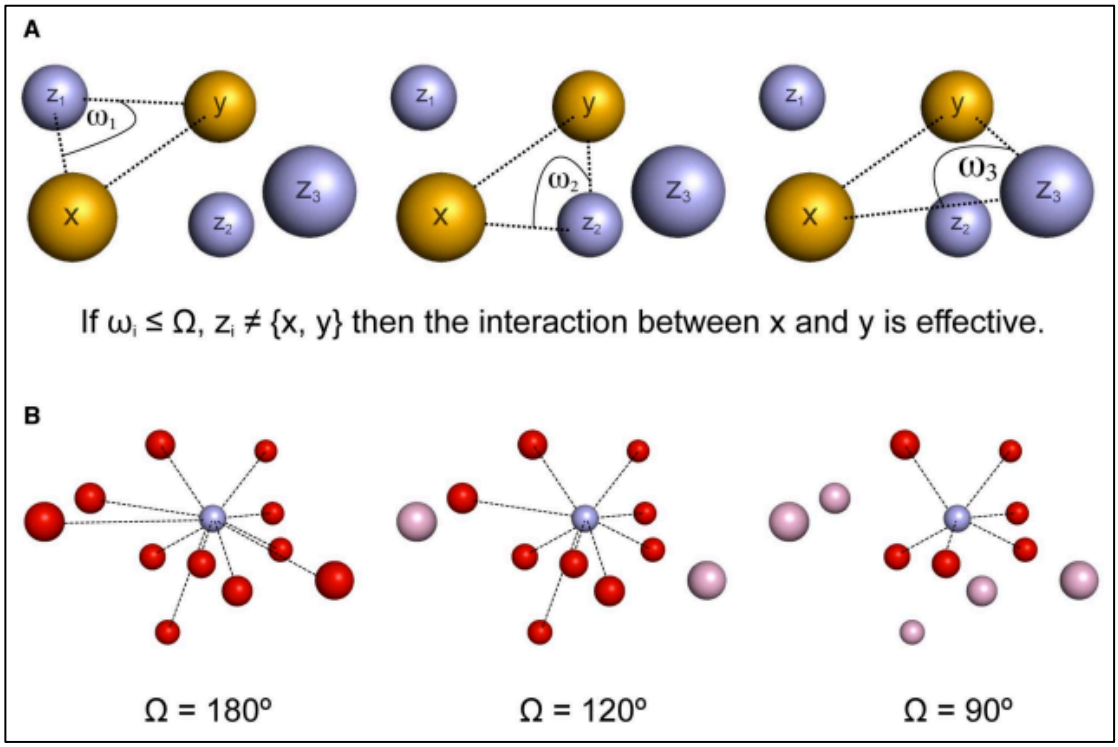


Figure 8: Polynucleotide-Protein-Interaction. *A* shows, which angles *ω* are calculated to check if the interaction between atoms *x* and *y* is shielded by one of the atoms z_i, where x and y belong to polynucleotide and protein, respectively. *B* visualizes the impact of different *Ω* on the number of effective interactions between polynucleotide and protein.

Dataset of DNA-protein interactions

The dataset describing DNA-protein interactions, also called the positive cases, is extracted from the Protein-DNA Interface Database (PDIDB) in the current release of 26.04.2010 (T. Norambuena 2010). The complete database contains 922 entries from the Protein Data Bank (PDB) (Helen M. Berman 2000) with a minimum resolution of 2.5 Å solved by X-ray crystallography. An additional

restraint by the authors of the database is that the protein-DNA complexes must contain double strand DNA. For the dataset used in this work, the identifiers were extracted.

These identifiers were then processed to determine which protein residues bind DNA/RNA and which do not. The effective interactions were determined using a tool developed by Shen Wei (Shen Wei 2012). Interaction was calculated with a distance cutoff lower than 5 Å and a shielding angle Ω of 90°.

Several processing steps and restrictions have to be added to make the original dataset usable as input to train the machine-learning method. First of all, a minimum length of 45 residues is requested to exclude short proteins, which would make the sliding window approach inefficient. Secondly, redundancy is removed by using UniqueProt (S. Mika 2003) with an HVAL of zero to remove overrepresented families or protein classes. HVAL is a value describing the similarity between two sequences using their alignment. The benefit of using the HVAL instead of just the fraction of identical aligned residues is, that it takes the length of the alignment into account. An HVAL of zero guarantees that the remaining proteins are dissimilar enough to significantly decrease the possibility of predictions by homology inference (Figure 9). The final dataset of DNA-protein interactions consists of 144 protein chains, which are binding DNA.

$$HVAL(L, PID) = PID - \begin{cases} 100 \; for \; L \leq 11 \\ 480 * L^{-0.32*\left\{1+\exp\left(-\frac{L}{1000}\right)\right\}} \; for \; L \leq 450 \\ 19.5 \; for \; L > 450 \end{cases}$$

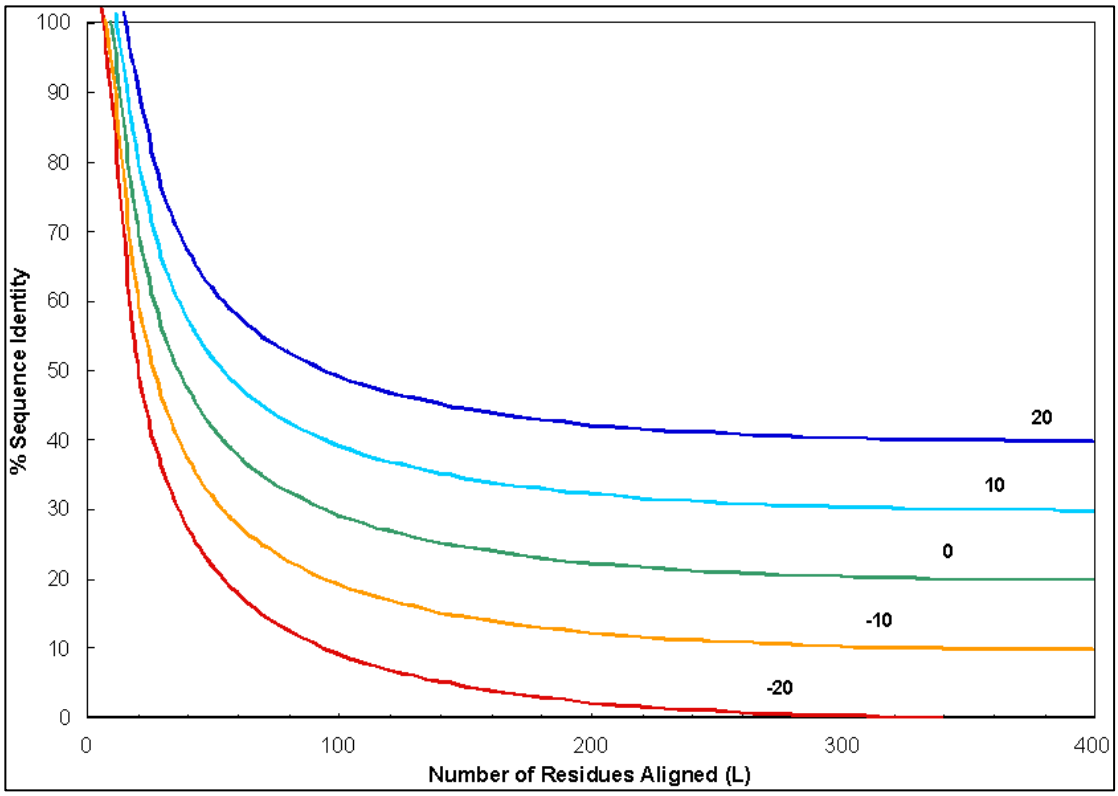


Figure 9: The plot gives an overview about the relation of the behavior of the HVAL dependent on the length of the alignment and the percent sequence identity. At the used threshold of zero, an alignment with the length of 200 can contain up to 25% identical residues.

Dataset of RNA-protein interaction

The RNA-binding proteins were extracted in a similar way to the DNA-binding ones. As a data source the full Protein-RNA Interface Database (PRIDB) was used, which provides a curated set of proteins binding RNA. The database version RB1179 contains 1166 PDB identifiers, but compared to the PDIDB the restrains are not as strict. Proteins, which are present in this database may not be solved by X-Ray crystallography, but also by nuclear magnetic resonance spectroscopy (NMR) resulting in multiple models per protein identifier. Due to this no statement about the resolution of the structures can be made.

Since the aim is to distinguish between RNA- and DNA-binding proteins, the datasets have to be comparable. To achieve this, the criteria used for the DNA-binding dataset, namely the solution by X-Ray crystallography and the resolution threshold of 2.5 Å, have been applied additionally to the RNA-binding data.

Utilizing those restraints, the redundancy reduction via UniqueProt and the extraction of effective interactions, 102 protein chains remain in the RNA-binding dataset.

Negative/Non-Binding Data

To improve the prediction quality with respect to distinguishing between proteins that bind polynucleotides and those, which do not, a dataset of non-binding proteins, the negative set has to be created.

As a starting point for this dataset SwissProt (T. U. Consortium 2011), a database containing manually annotated and reviewed protein sequences, is chosen. Most of the entries in this database are annotated using the Gene Ontology (GO) (T. G. Consortium 2000). The Gene Ontology is a directed acyclic graph, which itself consists of three ontologies namely "molecular function", "biological process" and "cellular component". As easy as it is to find proteins which bind polynucleotides using the GO annotations, the proceedings in the reverse order have to be much more restrictive, because there is no GO-term named "protein not binding DNA or RNA". To extract the negative dataset with a high certainty, only proteins in SwissProt, which have an annotation in every of the three sub-ontologies, are used. Additionally, those annotations are required to be flagged with an experimental evidence code. Thus, GO annotations with an evidence code other than those listed in the Table 1 are not accepted.

Experimental Evidence Code	Explanation
EXP	Inferred from Experiment
IDA	Inferred from Direct Assay
IPI	Inferred from Physical Interaction
IMP	Inferred from Mutant Phenotype
IGI	Inferred from Genetic Interaction
IEP	Inferred from Expression Pattern

Table 1: The experimental evidence codes in SwissProt/UniProt and their meaning.

Having the annotation of those three ontologies with experimental evidence codes suggests a well-known and –reviewed protein. To exclude proteins binding polynucleotides from the dataset, GO-terms not valid for the negative dataset are defined in the following paragraph. Additionally, the full names of the GO-terms were searched via regular expressions to limit the results even more.
The GO term to exclude/identify polynucleotide binding proteins that is set highest in the Gene Ontology graph is “nucleic acid binding” (GO:0003676). As the extraction method also takes the children of the unwanted terms into account, it also includes the terms for “DNA binding” (GO:0003677) and “RNA binding” (GO:0003723) as well as other terms like “DNA/RNA hybrid binding” (GO:0071667). Furthermore, all GO terms containing either of the words “DNA” or “RNA” in their names are excluded as potentially polynucleotide-binding proteins.

Additionally, the sequences of all proteins in the negative dataset have a minimal protein length of 45 residues as in the positive datasets.

After applying the restraints detailed above, 8366 proteins remain in the dataset, which decreases to 656 proteins after the redundancy reduction using UniqueProt with an HVAL of zero.

To ensure a clean dataset, the HVAL of zero is again applied to remove proteins out of the negative dataset, which are too similar to those in the positive one. Every protein of the positive sets (DNA- and RNA-binding) was checked against the negative set using BLAST (Stephen F. Altschul 1997). If the HVAL was above the threshold of zero, the protein was removed from the negative set. Although a redundancy reduction using the whole gathered data would result in more proteins in total, it is important to keep as many positive examples as possible, because the positive dataset is relatively small. Finally, the negative set contains 464 proteins. Table 2 shows the number of protein chains and residues in each set, and the distribution of polynucleotide binding to non-binding residues.

Dataset:	Binding Proteins:	Non-binding Proteins:	Total Proteins:	Binding Residues:	Non-binding Residues:	Total Residues:
DNA-binding:	144	0	144	3474	20480	23954
RNA-binding:	102	0	102	3408	14046	17454
Non-binding:	0	464	464	0	57588	57588

Table 2: The created datasets and their distribution of binding and non-binding cases.

Features of the prediction

A property, which helps the predictor to distinguish between the positive and the negative class, is called a feature. This property can for example be extracted by looking at the residue in a specific position, which has certain characteristics like its charge or mass. Features can further be split into local and global features. Global ones describe properties of the whole protein, like the amino acid composition or its length. Local features on the other hand are specific to each residue in the protein.

Sequence Based

Sequence based features are those, which can in principle be directly derived from the protein's sequence.

The most promising local feature is the **evolutionary profile**, which is used by many sequence based prediction methods. To receive it, HHblits (Michael Remmert 2012) is used to find homolog sequences for the query protein in a redundancy reduced Uniprot database (T. U. Consortium 2011). The result is a profile compiled out of the multiple-alignment of the found sequences, more exactly the position specific scoring matrix (PSSM) (Stephen F. Altschul 1997). This profile gives a probability for every amino acid out of the 20 possible at each position in the protein. Basically this gives information about the

conservation or diversity of amino acids at a specific position. The resulting values are normalized by the sigmoid function (Figure 10), which is a special case of the logistic function. Two additional input units are the last two columns of the PSSM file: the information per position and the relative weight of gapless real matches to pseudo counts.

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

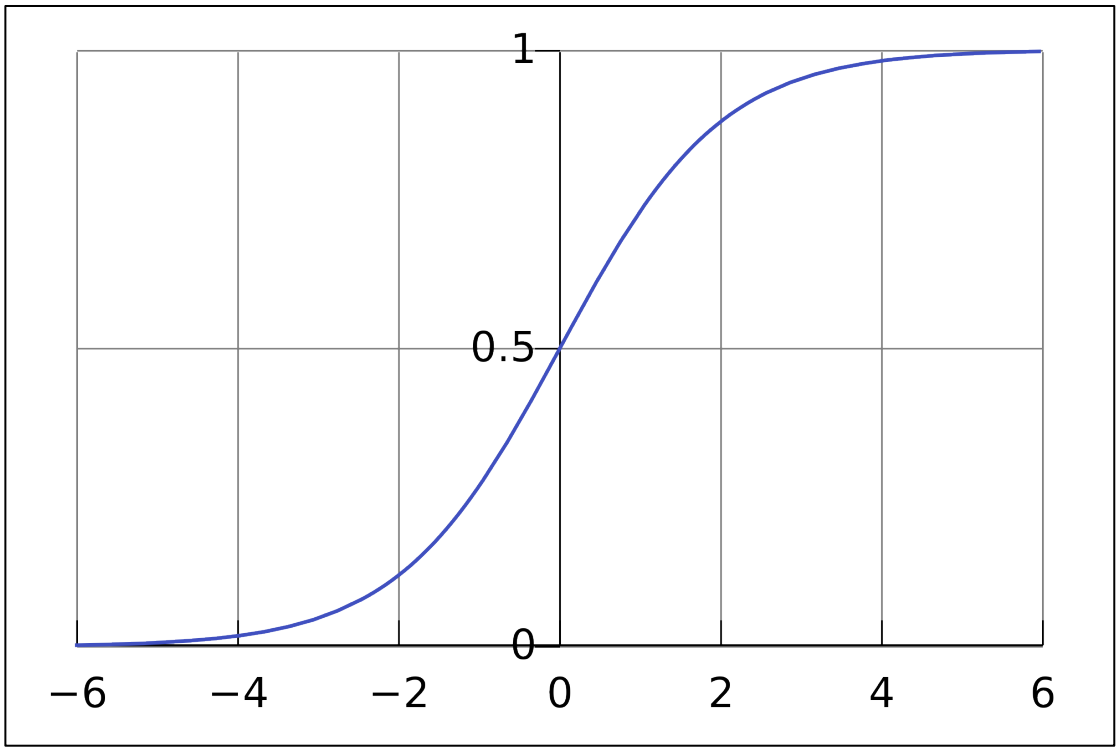


Figure 10: The sigmoid function, which is used to normalize the PSSM features on the one hand, and serves as the threshold function for the perceptrons on the other. Normalization means, that values exceeding the interval of 0 to 1 are transformed into this range.

Because a sliding window approach is used, it is necessary to provide the machine-learning algorithm with the information if a residue is existent in the protein sequence, or if it is a virtual residue to fill the inputs at the N- and C-terminal of the protein. This identification of the real vs. the virtual residues is called the **in-sequence-bit**, set to 1 and 0, respectively.

Other local features are **mass, volume, hydrophobicity, charge, polarity, the tendency to break alpha-helices and c-beta branched amino acids**. See Table 3 for the used values.

Amino acid	Mass	Volume	Hydrophobicity	Charge	Polarity	Helix-breaker	C-beta
A	0.109	0.170	0.700	0.5	0	0	0
R	0.767	0.676	0.000	1	0	0	0
N	0.442	0.322	0.111	0.5	1	0	0
D	0.450	0.304	0.111	0	1	0	0
C	0.357	0.289	0.778	0.5	0	0	0
Q	0.550	0.499	0.111	0.5	1	0	0
E	0.558	0.467	0.111	0	1	0	0
G	0.000	0.000	0.456	0.5	0	0	0
H	0.620	0.555	0.144	1	1	0	0
I	0.434	0.636	1.000	0.5	0	0	1
L	0.434	0.636	0.922	0.5	0	0	0
K	0.550	0.647	0.067	1	1	0	0
M	0.574	0.613	0.711	0.5	0	0	0
F	0.698	0.774	0.811	0.5	0	0	0
P	0.310	0.314	0.322	0.5	0	1	0
S	0.233	0.172	0.411	0.5	1	0	0
T	0.341	0.334	0.422	0.5	1	0	1
W	1.000	1.000	0.400	0.5	0	0	0
Y	0.822	0.796	0.356	0.5	1	0	0
V	0.326	0.476	0.967	0.5	0	0	1
X (other)	0	0	0	0	0	0	0

Table 3: Local features of amino acids. For each of the 20 amino acids, mass, volume, hydrophobicity, charge, polarity, the tendency to break alpha-helices and c-beta branched amino acids is given. Amino acids are listed in 1-letter code.

Two features indicate the position of the residue in the protein, which are the residue's **distances** D **from the N- and/or the C-terminus**. This property is encoded in a 4-unit feature using the following formula.

$$Input\ to\ unit\ i = 1, if\ D \geq 2^{i-1} * 10, and\ \frac{D - 2^{i-2} * 10}{2^{i-1}} * 10-, for\ i = 1 -\ 4$$

The **amino acid composition** of a protein is a global feature and consists of 20 positions according to the available amino acids. For each position, the number of occurrences of the amino acid corresponding to this position is counted and divided by the total number of residues in the protein.

Another global feature, which can be used, is the **length** N of the protein. Because the inputs for the machine-learning algorithm have to be scaled, the length was converted into a 4-unit feature.

$$Input\ to\ unit\ i = 1, if\ N\ \geq\ 2^{i-1} * 60, and = \frac{N - 2^{i-2} * 60}{2^{i-1}} * 60, for\ i = 1 -\ 4$$

PredictProtein

As opposed to the sequence-based features, there are those extracted from other sequence-based prediction tools. PredictProtein offers a huge set of such predictions, which are automatically run after submitting a query, like secondary structure, solvent accessibility, disordered regions and so forth (B. Rost 2003). Such predictions can contain additional properties, the machine-learning method cannot identify directly.

Reprof is a tool predicting secondary structure in three states (alpha-helix, beta-sheet and other), as well as the solvent accessibility for each residue. The secondary structure is converted into a 4-state feature with one bit for each state. A reliability index indicates the certainty of the prediction. Two additional input positions make the relative solvent accessibility available to the predictor, the value itself, and reliability index.

PROFbval (A. Schlessinger 2005) predicts residue mobility. The used feature is encoded by three positions. One for rigidity, one for flexibility, and one for the reliability of the prediction calculated by subtracting one value from the other.

Disorder is implemented as a two-bit input feature by **Metadisorder** (Avner Schlessinger 2009) predictions. The states can either be disordered, leading to a 1,0 input, while not disordered regions result in a 0,1 vector.

COILS (A. Lupas 1991) is a predictor for coiled-coils, meaning that two alpha helices are coiled together. The feature is encoded with 7 bits, depending on the length of the coiled-coil.

Another feature supplied by PredictProtein is the prediction of protein-protein interactions by **ISIS** (Yanay Ofran 2007). The output of this program is transformed into a 7-state feature, with each bit describing a different binding mode. If a residue is predicted as non-binding, all bits are set to zero.

Sliding Window Approach

Early sequence-based prediction methods in the field of bioinformatics took only the residue itself into account for which the prediction is done. By using this approach, none of the information about the local surrounding of the predicted residue is taken as input to improve prediction quality. To handle this issue, a sliding window approach is used in this work.

The window can have any uneven number larger or equal to one as its size *w*. Using a window size of *w*=1 is the same as using no window at all, since it considers only the central residue *i* for which the prediction is made. If the window size is increased to *w=5*, the input for the prediction of the central residue *i* is influenced by the feature values of the surrounding four residues *i-2*, *i-1, i+1* and *i+2*, plus the feature values of the central residue *i* itself.
Although this approach works the most parts of the protein, the treatment of the C- and N-terminal is special. If the central residue *i* is at the beginning of the sequence, e.g. *i=1* and using a window size *w=5*, the beginning of the window *i-2*

is at the non-existent position i=-1 (assuming the protein sequence starts at the iterator 1) . Since the machine-learning method expects a fixed number of input values for each residue to be predicted, those non-existent positions are filled with zeroes as feature values, indicating that these positions are outside the sequence's boundaries.

Additionally, for an easier distinction, the previous mentioned "in-sequence-bit" is added, which is set to 1 for every position inside the sequence, and to 0 for those outside. Therefore, the input to the machine-learning method is the concatenation of the feature values from all residues inside the window of the given length (Figure 11).

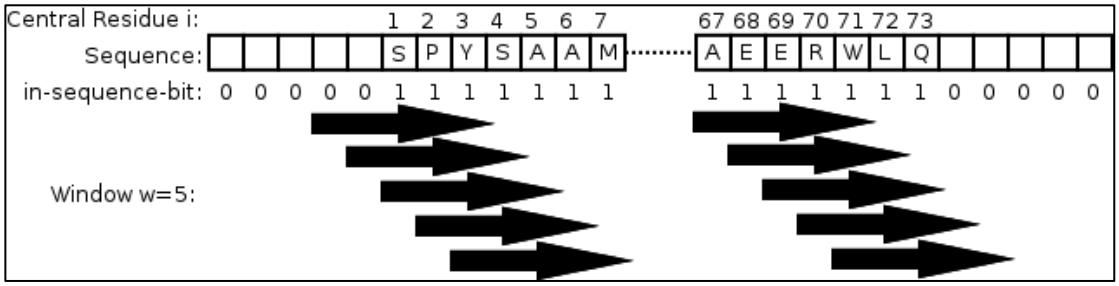


Figure 11: An example of how the window (indicated by the arrows) slides through the protein sequence.

Training Parameters

Cross-Validation

To evaluate the best parameter and feature combinations, a 5-fold cross-validation is used. Cross-validation is a method to compensate for the low amount of available data, making it possible to evaluate the performance of a method or parameter/feature combination without holding back some of the data for validation and testing, which could otherwise be used for training.

For the 5-fold cross-validation, the dataset is split into five equal-sized parts, *n* = 1 to 5. In the fold *n*, the parts *n* to *n+2* are taken for training the model, which means the data points included in those parts are presented to the neural-network learning algorithm. The part *n+3*, namely cross-training data, is used on the one hand to determine when to stop training, i.e. to find the point where the model is fitted well enough to the data without being over-fitted, and on the

other hand to evaluate the performance between the models trained with different parameter/feature combinations to select the best one. The *n+4* part of the data is for testing the final performance of the beforehand selected model. The average performances on the testing data of all five folds add up to the final performance of the predictor. A schematic and the datasets in folds *n* used here are depicted in Figure 12 and Table 4.

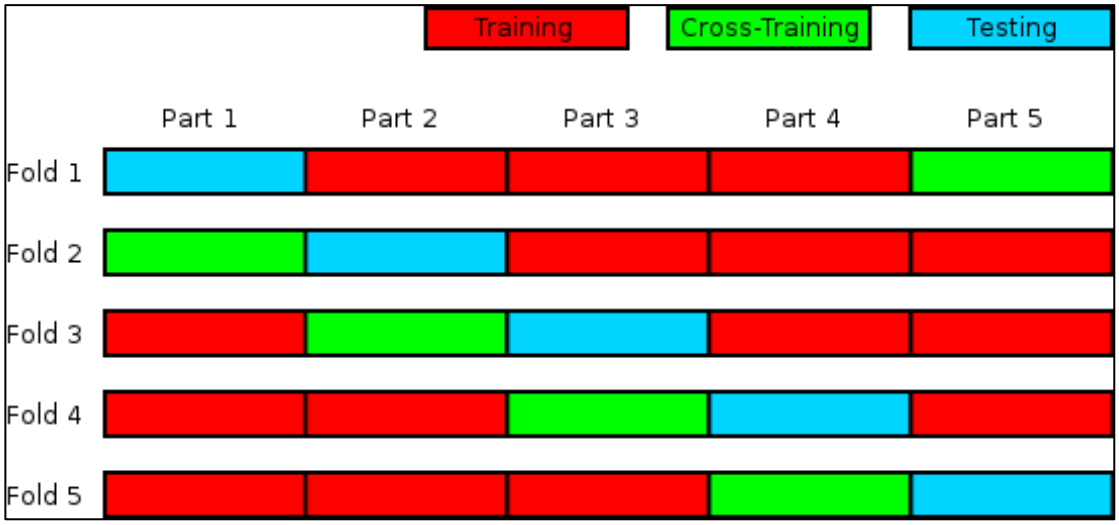


Figure 12: The split of the dataset in five parts resulting in five folds. The color indicates, which part is used either for training (red), cross-training (green) and testing (blue).

Part 1	Part 2	Part 3	Part 4	Part 5
ACBP_RAT	4EBP1_RAT	ACPM_MYCTU	APC14_SCHPO	A62F_DROME
ACM1_YEAST	APOA2_HUMAN	AMELX_MOUSE	APEL_RAT	ABP1_PIG
AGRP_HUMAN	ATNG_RAT	AP1S1_YEAST	APOC2_HUMAN	AHPD_MYCTU
AHP5_ARATH	ATX1_YEAST	APOC1_HUMAN	APOC3_HUMAN	AL5AP_RAT
ANFB_RAT	BET1L_HUMAN	ATPJ_YEAST	AR6P1_MOUSE	APTH1_YEAST
ATL62_ARATH	BNI3L_HUMAN	B2L11_HUMAN	CCL3_RAT	ATIF_YEAST
ATRAP_MOUSE	BURS_DROME	BKI1_ARATH	CD24_HUMAN	ATP5H_ARATH
AURB_CHLAA	CCW14_YEAST	BL1S1_HUMAN	CD70_HUMAN	BH135_ARATH
AUX1_ARATH	CHMU_MYCTU	CBLN4_MOUSE	CNIH2_RAT	BL1S3_HUMAN
AVE_DROME	CKLF_HUMAN	CCAP_MANSE	CO5_RAT	BRK1_ARATH
BEX3_MOUSE	CLE44_ARATH	CCKN_MOUSE	CXCL5_RAT	C42S2_HUMAN
CASP3_ARATH	COX17_YEAST	CLD2_MOUSE	CYFP1_BOVIN	CALC_MOUSE
CD59_RAT	CRGD_HUMAN	CP122_ARATH	CYTC_HUMAN	CART_RAT
CLV3_ARATH	CRIPT_RAT	CUE1_YEAST	DMS2_PHYBI	CCMD_ECOLI
COAA_DICDI	CWC16_YEAST	CUSF_ECOLI	DPM3_HUMAN	CD81_RAT
CPLX3_MOUSE	CWP2_YEAST	CXL10_RAT	EAF6_HUMAN	CDC25_ARATH
CX6B1_ARATH	CXXS1_ARATH	CYOC_ECOLI	EBNA5_EBVB9	CLC5A_MOUSE
CYOD_ECOLI	CYAY_ECOLI	DEGS_SCHPO	EF1B_DICDI	CLE2_ARATH
DAD1_YEAST	CYC4_THIRO	DHPR_RAT	ERD2_YEAST	COMI_DICDI
DIS2_DICDI	CYC7_YEAST	DROS_DROME	EXFAB_CHICK	COXM1_YEAST
DPOE3_HUMAN	DBH_MYCTU	EGL1_CAEEL	FTSB_ECOLI	CRPAK_HUMAN
DPY30_HUMAN	DSS1_CAEEL	ELF4_ARATH	FTSN_ECOLI	CT54_CONLT

DYL1_DROME	DYLT1_BOVIN	EPGN_HUMAN	G0S2_HUMAN	CTR4_SCHPO
EBP_RAT	ERI1_YEAST	EPT1_YEAST	GFRP_RAT	DAD2_YEAST
GCR1_ARATH	ESPD_MYCTU	FDNI_ECOLI	HK1_MYCTU	DPH3_HUMAN
GLRX1_YEAST	ESXA_MYCTU	GBG2_BOVIN	HLPDA_HUMAN	DPM2_RAT
HDBP1_MOUSE	FABP6_RAT	GBGE_DROME	HSP12_YEAST	EMD_HUMAN
HRK_HUMAN	FCERG_MOUSE	GHRL_HUMAN	IDH3A_PIG	EMRE_ECOLI
ISD11_YEAST	FSTL3_MOUSE	HCN1_SCHPO	IKP1_PHYSA	ERG28_YEAST
LACC1_CERUI	GLC8_YEAST	HIP26_ARATH	IL11_MOUSE	ERV2_YEAST
LSM10_HUMAN	GLUC_RAT	IDH3G_PIG	IL2_RAT	FXYD3_MOUSE
MED20_SCHPO	GOT1B_HUMAN	IL12B_MOUSE	KCNE1_HUMAN	GBG_YEAST
MGST1_MOUSE	HATA_DICDI	LITAF_HUMAN	ML328_ARATH	GRS14_ARATH
MPSIN_BOOMI	HNT1_YEAST	LYS1_CRAVI	NEDD8_CAEEL	HOT13_YEAST
MSRB2_ARATH	HPA3_YEAST	MCP_RAT	NEU2_RAT	IL15_MOUSE
NAPD_ECOLI	HPEP_AEDAE	MD2L2_XENLA	NPFF_RAT	JTB_HUMAN
NPF_DROME	IFNB_MOUSE	NACA_YEAST	ORCH_ORCCA	LEP_RAT
ORML3_HUMAN	IGF1_RAT	NAUT_NAUMA	OSTP_MOUSE	LPQH_MYCTU
OST2_YEAST	ILVN_ECOLI	NDE1_YEAST	PDE6D_DROME	MA6D1_MOUSE
PEX17_YEAST	ITBP1_HUMAN	NP7_RHOPR	PEN2_HUMAN	MAL_RAT
PFD1_YEAST	LACC2_CERUI	NTM1_YEAST	PGSA2_MYCTU	MED22_DROME
PMEI2_ARATH	LORI_HUMAN	NUOI_ECOLI	PIS1_ARATH	MTGA_ECOLI
PPT2_YEAST	LST7_YEAST	OSTCN_RAT	PLRKT_MOUSE	NC2B_HUMAN
PRIMA_MOUSE	MES1_SCHPO	PEM2_YEAST	PPLA_RAT	NIPA2_MOUSE
PT100_YEAST	MMGT1_MOUSE	PHLIP_ANUPH	PRA1_YEAST	NPY_RAT
QCR6_RAT	MPT53_MYCTU	PONA_DICDI	PROF2_MOUSE	NTAQ1_MOUSE
RL29_YEAST	MRAP2_HUMAN	PREY_HUMAN	RAMP2_HUMAN	OSPG_SHIFL
RPN13_YEAST	MRP10_YEAST	PRRP_RAT	RL101_ARATH	PA2GA_HUMAN
RS25_RAT	MSCL_MYCTU	SIP18_YEAST	RL121_ARATH	PAIP2_HUMAN
S10A4_HUMAN	MT1G_HUMAN	SLG1_YEAST	RL7A_YEAST	PCNP_HUMAN
S10AE_HUMAN	MYPR_MOUSE	SLIB_MOUSE	RM49_YEAST	PGC1_YEAST
SC6B2_YEAST	NIC1_SCHPO	SMR1_RAT	RS21_ECOLI	PRP38_YEAST
SCXA_MESMA	NRT31_ARATH	SNAT_HUMAN	S10A7_HUMAN	PTHR_HUMAN
SFT1_YEAST	OB76A_DROME	SNO1_YEAST	SAP18_ARATH	PTPS_MOUSE
SNA3_YEAST	PEN3A_LITVA	SPC19_YEAST	SCAB_ORYRH	QVR_DROME
SSBP_HUMAN	RBX2_MOUSE	SRTD3_HUMAN	SDHF2_HUMAN	RGF1_ARATH
STG1_SCHPO	RM51_YEAST	STE14_YEAST	SED1_YEAST	RL22B_YEAST
TEBP_RAT	RPC9_MOUSE	STHY_ARATH	SKA2_HUMAN	RM52_BOVIN
TIM10_HUMAN	RT24_BOVIN	SWM1_YEAST	SOSD1_MOUSE	RM55_BOVIN
TIM12_YEAST	SKP1_HUMAN	TDGF1_HUMAN	SPR1A_HUMAN	RPIB_MYCTU
TIM13_YEAST	SPC34_YEAST	TIM14_YEAST	SRX1_YEAST	RT63_MOUSE
TSG_DROME	SSEB_SALTY	TIM16_YEAST	STX8A_DICDI	SAP3_MOUSE
TTS1_SCHPO	SVP26_YEAST	TNFL6_RAT	SUMO2_ARATH	SAP5_ARATH
V15A2_AEDAE	SYUA_HUMAN	TPPC4_MOUSE	TELT_HUMAN	SC61G_YEAST
VMA22_YEAST	TM204_MOUSE	TRIA1_HUMAN	TIM8_YEAST	SEC22_YEAST
VTU4_DROME	URE3_MYCTU	TSC3_YEAST	TIM9_HUMAN	SMRP1_MOUSE
WRBA_ECOLI	UTER_RAT	UPK1A_BOVIN	TIP1_YEAST	SNE_ARATH
XCL1_MOUSE	VAMP8_RAT	YE0B_SCHPO	TRXM2_ARATH	SPD1_SCHPO
Y098_MYCTU	VASOT_HYBBI	YL194_YEAST	UBE2W_MOUSE	TIM21_YEAST
YIF1_YEAST	VKOR1_RAT	YPI1_YEAST	URE2_MYCTU	TMM97_HUMAN
YOP1_SCHPO	VPS51_YEAST	YSF3_YEAST	VA0E_YEAST	TPX_ECOLI
ZAPB_ECOLI	VSP1_GLOBR	Z600_DROME	VATO_YEAST	UCN2_RAT
ZIM17_YEAST	ZEO1_YEAST	ZYM1_SCHPO	VCY2_HUMAN	ZN363_HUMAN
dna_1ckq:A	dna_1am9:C	dna_1awc:A	dna_1awc:B	dna_1dc1:B
dna_1dmu:A	dna_1ckt:A	dna_1b95:A	dna_1b3t:A	dna_1emj:A
dna_1gd2:E	dna_1d3u:B	dna_1bdt:B	dna_1cw0:A	dna_1hcr:A
dna_1je8:E	dna_1esg:B	dna_1bl0:A	dna_1dp7:P	dna_1hlv:A
dna_1jj4:B	dna_1eyu:A	dna_1cez:A	dna_1e3o:C	dna_1ipp:A

dna_1kbu:A	dna_1fiu:A	dna_1dfm:A	dna_1fjl:C	dna_1kx5:D
dna_1kx5:C	dna_1h89:C	dna_1gxp:A	dna_1gu4:B	dna_1mnm:A
dna_1oup:A	dna_1j3e:A	dna_1iaw:B	dna_1ign:A	dna_1mus:A
dna_1qaj:A	dna_1mj2:B	dna_1j1v:A	dna_1kx5:A	dna_1n6j:A
dna_1qne:B	dna_1nfk:A	dna_1jft:A	dna_1lq1:C	dna_1nkp:A
dna_1qpi:A	dna_1nvp:D	dna_1ku7:A	dna_1oe4:A	dna_1pp7:U
dna_1sa3:A	dna_1ozj:A	dna_1kx5:F	dna_1puf:A	dna_1qrv:A
dna_1t2t:A	dna_1puf:B	dna_1l3l:B	dna_1r71:B	dna_1r7m:A
dna_1zs4:A	dna_1rxw:A	dna_1nlw:A	dna_1r8e:A	dna_1rh6:A
dna_2bnw:B	dna_1skn:P	dna_1yf3:A	dna_1sxp:B	dna_1rio:A
dna_2efw:A	dna_1trr:A	dna_1yrn:B	dna_1w0u:A	dna_1rm1:C
dna_2fcc:A	dna_1vrl:A	dna_1zrf:A	dna_1zme:C	dna_1sfu:A
dna_2fkc:A	dna_1wtr:A	dna_2b9s:B	dna_2bop:A	dna_1tc3:C
dna_2h27:A	dna_2aor:B	dna_2dgc:A	dna_2hap:C	dna_1u8b:A
dna_2ihm:B	dna_2b9s:A	dna_2dnj:A	dna_2is6:A	dna_2ata:B
dna_2pi0:B	dna_2c6y:A	dna_2dp6:A	dna_2oaa:B	dna_2c9l:Y
dna_2ql2:B	dna_2fld:B	dna_2e1c:A	dna_2qhb:A	dna_2d5v:A
dna_2vs8:F	dna_2heo:A	dna_2ezv:B	dna_2ql2:C	dna_2etw:A
dna_2zkd:A	dna_2nll:B	dna_2h7h:B	dna_2ve9:F	dna_2gih:A
dna_3c2i:A	dna_2r1j:R	dna_2i13:A	dna_2yvh:B	dna_2rbf:B
dna_3e6c:C	dna_2r9l:A	dna_2isz:C	dna_3bep:A	dna_3btx:A
dna_3ere:D	dna_2z3x:A	dna_2ntc:A	dna_3bs1:A	dna_3coq:B
dna_3hts:B	dna_3cro:L	dna_2ofi:A	dna_3c25:A	dna_6pax:A
dna_9mht:A	dna_3f21:A	dna_2p0j:A	dna_3cvu:A	rna_1a34:A
rna_1dfu:P	rna_1ffk:G	rna_1uvm:A	rna_1jj2:V	rna_1feu:D
rna_1ffk:Z	rna_1h2c:A	rna_1vq8:A	rna_1ooa:A	rna_1ffk:U
rna_1gtn:M	rna_1jbr:A	rna_1vq8:B	rna_1vq8:V	rna_1jid:A
rna_1mji:A	rna_1knz:I	rna_1vq8:U	rna_1vqk:2	rna_2a8v:A
rna_1rpu:A	rna_1vq7:N	rna_1vqm:Q	rna_2bh2:A	rna_2anr:A
rna_1urn:C	rna_1vq8:S	rna_2asb:A	rna_2dlc:X	rna_2bs0:C
rna_1vq7:L	rna_1vql:I	rna_2b3j:C	rna_2pjp:A	rna_2g8h:A
rna_1vq7:O	rna_1zjw:A	rna_2hw8:A	rna_2vqe:I	rna_2qux:J
rna_1vq8:C	rna_2f8k:A	rna_2q66:A	rna_2vqe:J	rna_2uwm:B
rna_1vq8:T	rna_2i82:A	rna_2qkb:A	rna_2vqe:R	rna_2vqe:B
rna_1vq9:K	rna_2jea:A	rna_2vqe:H	rna_2xlk:A	rna_2vqe:F
rna_1vqk:1	rna_2vqe:K	rna_2vqe:P	rna_2xnr:A	rna_2vqe:G
rna_1vql:Z	rna_2vqe:Q	rna_2y8w:A	rna_3add:A	rna_2vqe:L
rna_1vqp:E	rna_3adl:A	rna_3cc2:P	rna_3d2s:B	rna_2vqe:N
rna_1yyk:B	rna_3agv:A	rna_3ex7:I	rna_3ks8:B	rna_2vqe:O
rna_2gxb:B	rna_3boy:C	rna_3hax:D	rna_3mdg:A	rna_2vqe:S
rna_2r8s:L	rna_3bsu:G	rna_3hsb:B	rna_3mj0:A	rna_2vqe:T
rna_2vqe:D	rna_3eqt:B	rna_3i5y:A	rna_3nmr:A	rna_2x1f:A
rna_2vqe:M	rna_3er9:B	rna_3oin:A	rna_3ovb:A	rna_3hjw:B
rna_3bsn:A	rna_3gpq:A	rna_3r2c:A	rna_3r9w:A	
rna_3o7v:X	rna_3hax:A			

Table 4: The identifiers contained in each of the five parts used for cross-validation after splitting the data. RNA binding proteins are prefixed with "rna_", while DNA binding proteins are prefixed with "dna_".

Training Stop Criterion

Normally, a trained model, which is both complex according to its parameters and trained long enough, would memorize the training data and therefore be over-fitted. To avoid this situation, the training has to be stopped at an earlier point, where the model sufficiently fits the training data, but is also applicable to data describing the same problem.

To achieve this, the previously mentioned cross-training data is used. After each training epoch, which means after all training samples have been presented to the learning algorithm, the model's performance is validated on the cross-training data. At first, the model performs better with each epoch on the training data as well as on the cross-training data as can be seen in Figure 13. However at some point, the performance on the training data increases further, while the performance on the cross-training data decreases. At this point, the training is stopped and the model performing best on the cross-training data is chosen to be the final model for this feature/parameter combination.

To avoid local maxima in cross-training data's performance, the training is continued for 10 further epochs looking for other, even higher maxima.

Figure 13: The behavior of the performance on the training and cross-training data during a neural network training run. While the performance on the training data increases until the data is memorized completely, the performance on the cross-training data raises to a maximum until the model generalizes well, then decreases.

Dataset Sampling

As described in the dataset section, the proportion of binding to non-binding residues is unequal. There are several basic methods to treat this imparity, which can in some cases improve the quality of the trained models. The simplest method is to leave the data as it is, which can be the best technique to keep the distribution fitted to the real world. One disadvantage could be, that the machine-learning algorithm used may forget or overlook the signals of the positive class, because it is underrepresented and overshadowed by the negative class. To handle this problem, undersampling or oversampling can be used.

Undersampling is achieved by taking only as many samples from the majority class, as are available in the minority class, which are in this case the positive or binding samples. The other way around is oversampling, where the samples from the minority class are duplicated until their count reaches the number of samples in the majority class.

Feature Selection

For the feature selection a fast and efficient greedy forward selection is used. It consists of several rounds: in the first round, all features are tested for their performance individually. The feature used by the best performing model is selected. In the next round, this feature is used along with all remaining features individually again. This routine is continued as long as the performance raises, which implies that the added feature provides additional signals to the machine-learning algorithm.

Benchmark And Validation Measures

Finding the optimal parameter and feature constellation requires different measures describing the performance of the created model. These measures are also used to compare the final method to already existent methods.

Each of the described measures highlights a different aspect of performance. Some measures only give a statement about one of the two possible classes, which are the positive or binding and the negative or non-binding class. Other measures attempt a combined score for the positive and negative class, to give a more general idea about the underlying performance. These measures can reflect the ability of the method to make precise predictions or the ability not to miss any of them.

Before describing the different measures, four terms have to be clarified, which describe the possible outcomes of a prediction, i.e. the predicted value, in relation to the real and correct one, called the observed value.

If the predicted value is positive and agrees with the observed one, it is called a true positive (TP). True negatives (TN) mean that the predicted and the observed value is negative. TP and TN specify correct predictions. However, the machine-learning algorithm also makes mistakes, resulting in false positive (FP) and false negative (FN) values. FNs are predictions, which are observed as

members of the positive class, but are predicted falsely as negatives. On the contrary FPs are observed as negatives but the predictor declares them as positive samples (Table 5).

		Observed value	
		Positive	**Negative**
Predicted value	**Positive**	TP	FP
	Negative	FN	TN

Table 5: Nomenclature of the four different prediction outcomes. The predicted value is the output of the predictor, while the observed value is the class label.

The first measure is the Q2, which gives an estimate about the overall performance of the predictor. The caveat is the dependence on the class distribution, meaning that predicting all samples as member of the majority class always gives rather good performance values.

$$Q_2 = \frac{TP + TN}{TP + TN + FP + FN}$$

Matthews's correlation coefficient is a measure, which takes both classes into account and can also be applied if the classes have different sizes. The value ranges from -1, representing total disagreement between the observation and the prediction, to 1, implying perfect predictions.

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}}$$

The mean squared error (MSE) is mainly used to select the best predictor since the neural network also uses it to improve the model.

$$MSE = \frac{1}{n} * \sum_{i=1}^{n} \frac{(p_+ - o_+)^2 + (p_- - o_-)^2}{2}$$

Where n is the number of samples, p and o the predicted and observed values derived from the original neural network output.

The precision (P) can be calculated for either the positive (P_+) or the negative (P_-) class. It gives a statement about how many of the samples are predicted, as members of the considered class are correct predictions.

$$P_+ = \frac{TP}{TP + FP}$$

$$P_- = \frac{TN}{TN + FN}$$

Recall (R) is also a measure evaluated for both classes (R_+ and R_-). It describes the fraction of samples covered correctly.

$$R_+ = \frac{TP}{TP + FN}$$

$$R_- = \frac{TN}{TN + FP}$$

The combination of precision and recall is called F1 measure (F1).

$$F1_+ = \frac{2 * P_+ * R_+}{P_+ + R_+}$$

$$F1_- = \frac{2 * P_- * R_-}{P_- + R_-}$$

The last measure described is the area under the curve (AUC) calculated out of the receiver operating characteristics (ROC). The main benefit of this measure is to be threshold independent. To calculate this value, the predictions are first sorted by their strength. Then two values, R+ and the false positive rate (FPR = FP / (FP + TN)), are calculated for each of these points, which is equivalent to

using every possible threshold. Finally, trapezoid fitting approximates the area under the curve produced by these points.

Neural Network Output

As mentioned earlier, the neural network used in this thesis has two output nodes, because two states, or classes, represent the binding prediction problem: binding and non-binding. These nodes, o1 for binding and o2 for non-binding, can range each from 0 to 1. This means in the training process, the pattern o1=1 and o2=0 is presented to the algorithm in the case of a binding residue, and o1=0 and o2=1 otherwise. As the trained neural network model will not be able to exactly predict those values, but their tendency to the desired observed output, a one-valued score is introduced using:

$$score = o_1 - o_2$$

This means, that for a very sure prediction towards the class “binding”, the score will be nearly 1, while the score for the class “non-binding” should approximate -1.

Another benefit of reducing the number of output values from two to one is the possibility of shifting the threshold for the predictions. For example, if only very precise binding predictions should be accepted, the threshold can be adjusted towards 1 (e.g. 0.7), resulting in an increase of precision for the binding class, at the cost of its recall. Stated differently, only very certain predictions of binding residues are accepted, at the cost of missing more binding residues, i.e. increasing the FN rate.

Threshold Selection

Using a threshold of 0 often results in a biased prediction in either the direction of a good recall or a good precision regarding the binding class. To balance this out, the precision and the recall are plotted against the score to decide at which score both of those measures are similarly high.

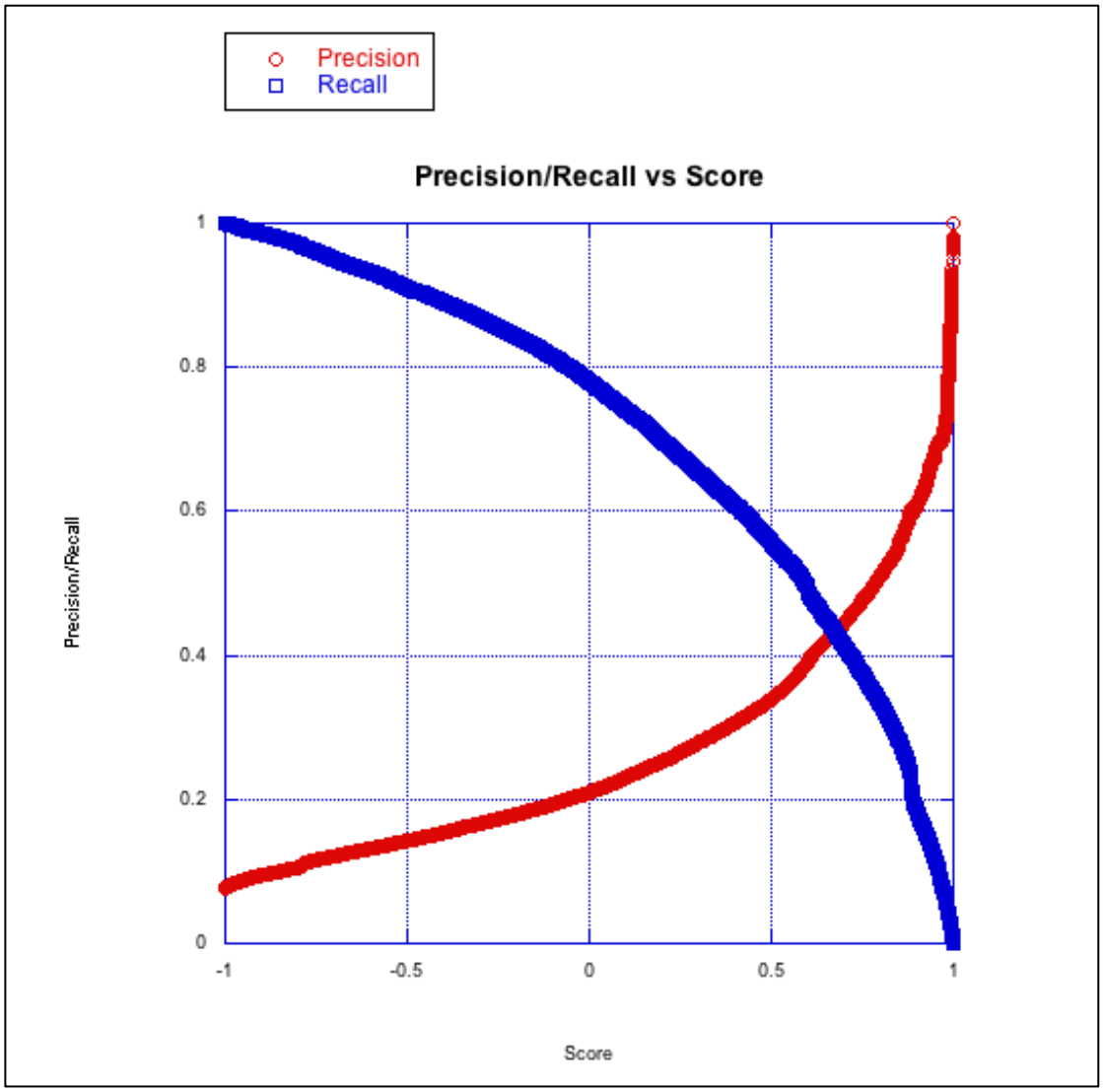


Figure 14: The plot visualizes the dependency of precision P and recall R if the threshold is shifted through the possible spectrum of the score. For a balanced prediction, a threshold can be chosen where the precision is at a similar value as the recall.

Assuming a score of 0 in Figure 14 the recall is fairly near 0.8, at the cost of a very low precision. If the score threshold is adjusted to about 0.7, where the recall and the precision lines meet, the recall decreases, while a significantly higher precision is gained. Using this method results in a more balanced prediction regarding precision and recall, in comparison to setting the threshold arbitrarily by hand.

From Residue To Protein-Based Prediction

The novelty of the prediction method in this work is to predict a proteins capability to bind DNA, RNA or none of them. To achieve this goal, the residue-based prediction has to be transformed into a protein-based one.

For the observed data, a protein is declared as binding a polynucleotide if at least one residue is an effective interacting residue as described in the interface definition section. Consequently, a protein where the method predicts a polynucleotide-binding residue should be a binding protein, too. However, since the residue-based prediction is not perfect, FPs would cause a high number of falsely positive predicted binding proteins.

Further investigations of the distribution of binding residues show, that they in fact are not correlated with the length of the proteins, but they tend to accumulate on the protein's sequence forming binding regions. To benefit from that circumstance, the proceeding is to score residues predicted as binding better if they have other binding residues near them. Because binding residues, which are far away in sequence can be close in 3D, binding residues appearing separate also contribute to the score, although not with such a high impact.

The final solution used in this thesis is a window approach, which incorporates clusters of binding residues for the scoring. Every residue in the protein receives a score that depends in the number of binding residues found in a window of 5 around the current scored residue, including itself. Thus, this value can range from 0, meaning no binding residue has been predicted in this window, to 5, implying all residues in the window are predicted as binding. After this step, the score from every residue is added up to the final score (Figure 15).

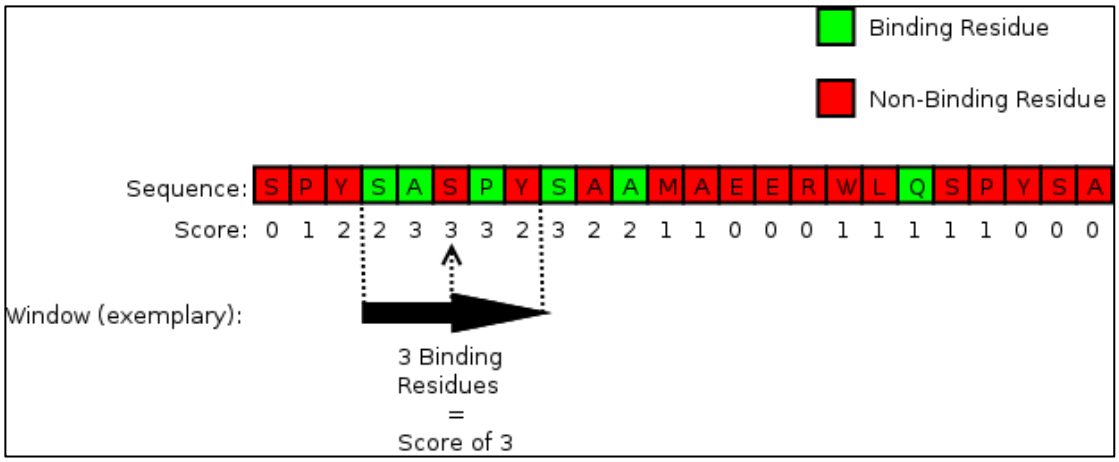


Figure 15: An example of how the per-protein score is calculated. In a window of five (indicated by the arrow), the residues predicted as binding are counted and add up to the score of the central residue of the window. This score is calculated for all possible windows. In the end, all these scores are summed up forming the per-protein score. The benefit of this calculation is to reward clusters of binding residues.

Training Workflow

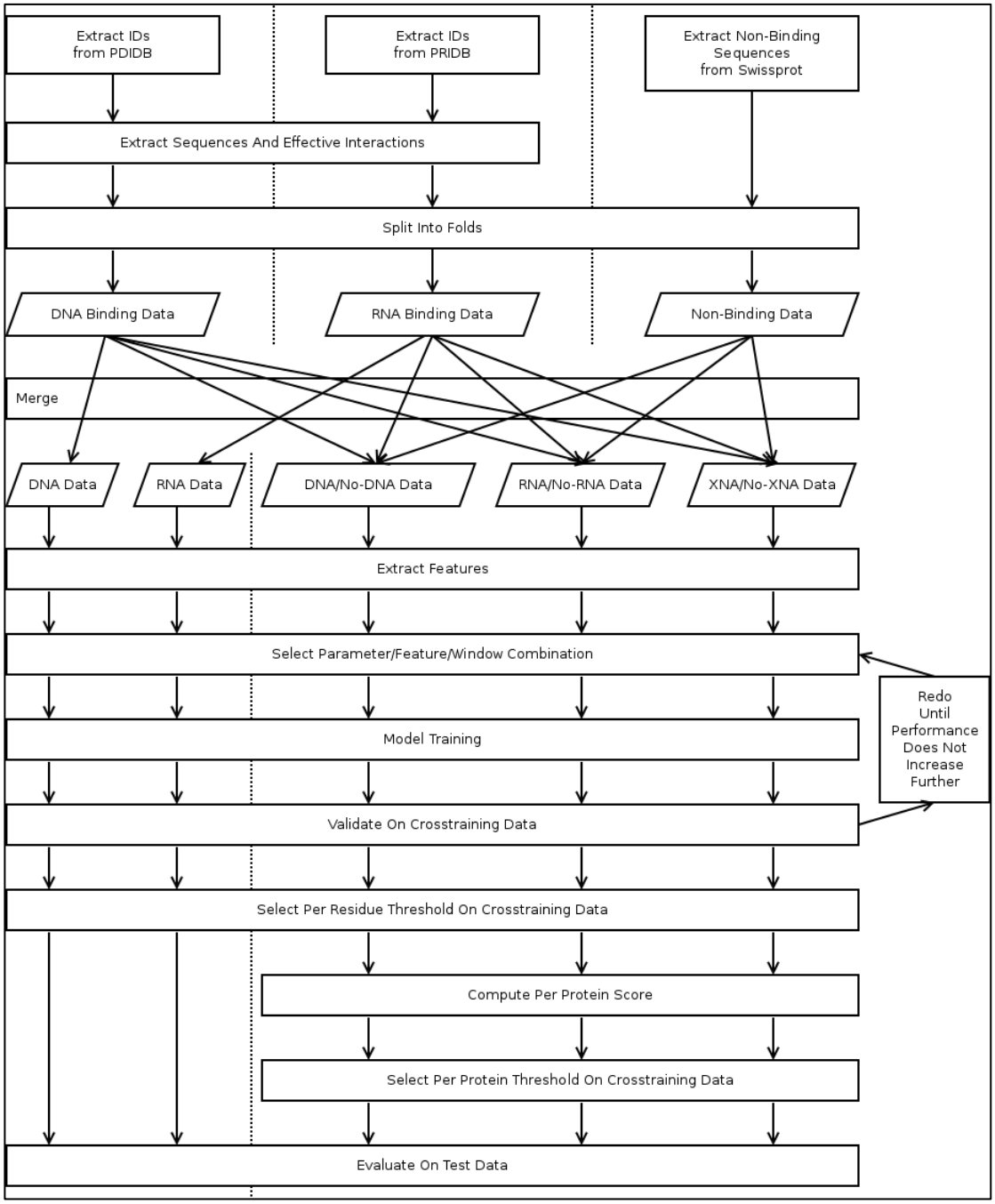


Figure 16: The flowchart of the training workflow. A detailed description can be found in the text.

The training of the method involves the interlocking of several neural network models and datasets, each tailored for its specific task in the whole training pipeline. The first steps consist of data gathering and the extraction of the target classes, in this case binding DNA, RNA or none of them. After this, these three data sets are split into the five parts for the cross-validation. Then, three clean datasets are created, each containing one of those target classes. The following

stages refer to all folds created by this split, it has to be noted that from this procedure on every fold is a self-contained system never influencing, or getting influenced by one of the other folds.
Because the machine learning method has to be capable to distinguish between different positive classes, those data sets are merged in different ways to create three further data sets, which, in fact, contain the same proteins, but their residues have different labels according to the current sub-problem, which has to be solved. One of those data sets has only the DNA binding residues labeled as binding. The second data set of those three has only the RNA binding residues labeled as binding, while in the third set the DNA as well as the RNA binding residues are the positive cases. As can be seen in Figure 16 there is also a clean DNA as well as a RNA binding set for a more precise residue-based binding prediction once a protein has been declared as polynucleotide binding.

After creating the features for each protein, the following step is the most computational intense one: the training of the neural network models. This training step is a loop, which consists of trying out different parameter and feature combinations, and assessing them on the cross-training data as long as no further performance increase can be achieved. Once the best model for each sub-classification part has been found, the prediction performance is balanced out by shifting the threshold for each neural network output so that precision and recall on the cross-training data are at a similar high level.

The last step to train is the threshold for the protein-based scoring, which is also done on the cross-training data targeting on a balanced prediction result.

Finally each part of the prediction method, including all models and thresholds, are evaluated on the test data of each fold, resulting in the ultimate prediction performance.

Prediction Workflow

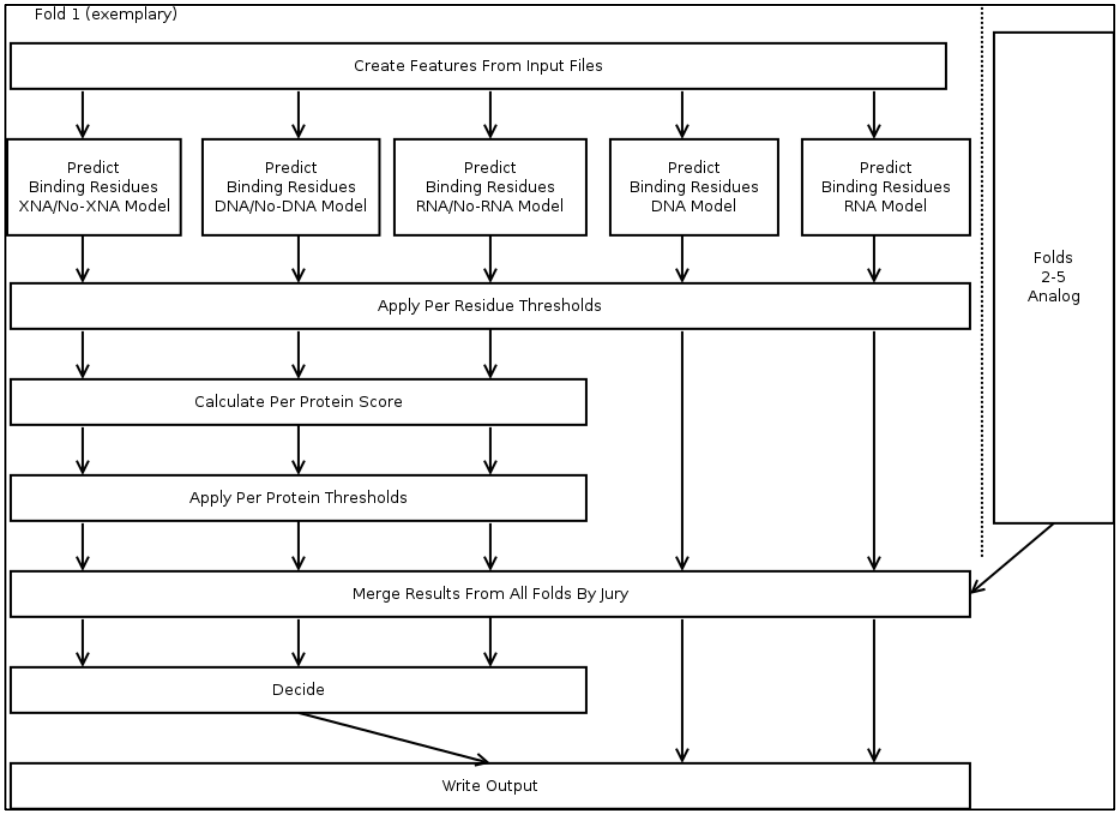


Figure 17: The flowchart of the prediction workflow. The detailed description can be found in the text.

For the final prediction method, the models and thresholds from the cross-validation are taken and incorporated into a prediction workflow (Figure 17). First of all, the features used by the neural network models are extracted from the input files. Those features then serve as input for the five neural networks, resulting in five different per-residue predictions to which the particular thresholds are applied. Three of the five models serve to discriminate between the different protein classes, so for the results of those, the per-protein score is calculated and again processed with the respective thresholds.

As this is done with the neural network models and thresholds of all five folds, the next step is to merge the folds' results to gain an ensemble prediction. For the per-protein prediction, which can only be 1 or 0 in each fold and class, this is done by a majority vote, meaning if the result from three folds is positive, the final result is positive, too. The final per-residue prediction is achieved by averaging over the score of each fold and residue.

The detailed voting strategy for the per-protein prediction is visualized in Table 6, which could also be illustrated by a decision tree. The first decision is whether the protein binds any polynucleotide (XNA) or not. If it does, it is tested whether it is more likely to bind DNA or RNA by assigning the class with the higher score. To avoid unsafe predictions, a protein can also be assigned to only the XNA but neither the DNA or RNA class, which happens if neither the DNA nor the RNA score reaches the class-specific threshold. It is however not possible to predict DNA or RNA binding without a prediction for polynucleotide binding.

XNA Prediction	**DNA Prediction**	**RNA Prediction**	**Prediction Result**
Yes	No	No	XNA
Yes	Yes	No	DNA
Yes	No	Yes	RNA
Yes	Yes	Yes	Not Possible
No	No	No	None
No	Yes	No	None
No	No	Yes	None
No	Yes	Yes	None

Table 6: The possible outcomes of the single predictors (XNA, DNA and RNA), and the resulting classification after combining them.

Results

The following chapter deals with the results of the beforehand trained neural network models and selected thresholds as well as the training itself. Apart from the section about selected features, it is divided into four parts according to the types of prediction systems involved in the final method beginning with the prediction of protein residues binding DNA, RNA or any polynucleotide at all (XNA). The next step is the transition from residue-based to protein-based prediction by presenting the residue-based neural network models' performance of those networks that were trained including the non-binding proteins. The last two parts are about the protein-based predictions and the differentiation of a protein's capability to bind DNA, RNA or none of them.

As the final classifier uses six prediction systems, only the performance plots for the positive, or binding class, in terms of precision and recall on the test set are shown. Further plots presenting also the negative class performance, and also different measures, e.g. the receiver operating characteristics curve (ROC) can be found in the annex/appendix.

The given thresholds are those, which lead to a balanced performance on the particular cross-training data in terms of equal precision and recall of the positive class.

Feature and Parameter Selection

To find out which features contain a signal helping the neural networks to distinguish between the different classes, a feature selection has been done. The approach was a simple forward selection done globally on all folds. The process of forward selection means that each trained neural network model was evaluated using the MSE with any additional feature that was not selected at this point. The added feature that brought the most increase in performance was selected and the selection process restarted with the remaining features. The term "globally" stands for a simplification that was made to the process.

Although an interaction between the training of neural network models of different folds is not designated, every feature that was one of the best performance candidates in each fold was also added to the selected features of the other folds. However, that does not apply for the parameter selection, which includes the number hidden units, the type of balancing and the window sizes. Table 7 shows the parameters that were tested during the parameter selection process. In addition to the already described balancing methods, it turned that many classificators performed best when averaging over the output values of the unbalanced, and the balanced network.

Parameter	**Tested Values**
Hidden Units	5, 10, 15, 20, 25, 30, 40, 50, 60, 70, 80, 90, 100
Balancing	None, Oversampling, Mixed
Window Size	9, 11, 13, 15, 17, 19, 21, 23, 25

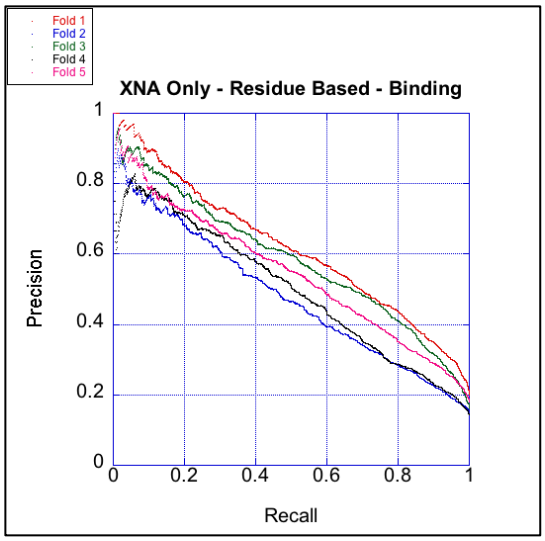
Table 7: The parameters that were tested during the parameter selection process.

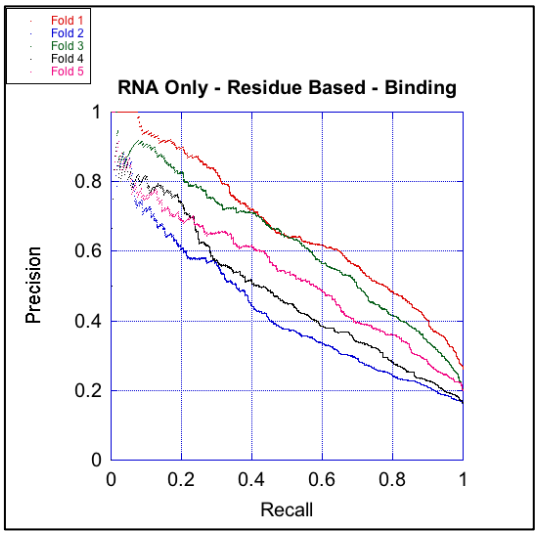
As a result, the average number of hidden units is 60, ranging from a minimum of 10 units to the maximum of 100. The number of networks used in the final predictor is 30. This number adds up from three networks for the residue-based prediction only and three networks for the residue-based prediction for the per-protein score calculation multiplied by five for the number of folds. Only one of those networks shows best performance when using oversampling while 17 networks prefer the mixed method. The remaining 13 networks are unbalanced. The window length has an average of 17, but the single network models utilize the full range of possible values from 9 to 25.

During the global forward selection, all features described in the methods section were selected except the disorder prediction, and the coiled-coils feature. The reason why the coiled-coils feature does not work could be founded in the encoding, because the ratio of one informative bit in a vector of seven is rather low.

Residue-Based Prediction

The residue-based predictions are interesting mainly if a protein is already classified as binding a polynucleotide. Residue-based means, that each residue in a protein is assigned a class label, which can either be binding or non-binding. A binding residue is defined by a certain proximity to nucleotide as described earlier as effective interaction. Thus, the following plots and performances are compiled based on the binding proteins referring to the actual case. The limitation to those cases, especially during the training, leads to an increase in residue-based performance. This is the case because the neural network is only confronted with residues inside binding proteins instead of being distracted by the signals occurring in non-binding ones. As the method aims mainly at protein-based predictions, the residue-based ones can be seen as an additional feature for further investigation of binding sites of the beforehand-classified proteins.





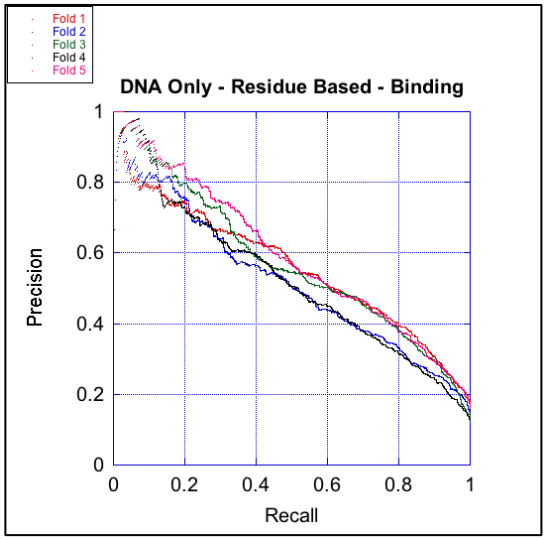


Figure 18: Precision plotted against the recall of the residue-based classifiers for XNA, DNA and RNA. Data is derived from the test parts of the 5-fold cross-validation and only the particular polynucleotide binding proteins are included.

XNA

	Threshold	P_+	R_+	FPR	P_-	R_-	Q2	MCC
Fold 1	0.04	0.66	0.43	0.06	0.87	0.94	0.84	0.45
Fold 2	-0.03	0.45	0.54	0.12	0.91	0.88	0.83	0.38
Fold 3	0.00	0.57	0.54	0.08	0.91	0.92	0.86	0.47
Fold 4	-0.35	0.52	0.49	0.07	0.92	0.93	0.86	0.43
Fold 5	-0.38	0.48	0.61	0.15	0.91	0.85	0.81	0.42
Average		**0.54**	**0.52**	**0.10**	**0.90**	**0.90**	**0.84**	**0.43**

Table 8: The precision and recall values, the Q2 score, the FPR and the MCC of the residue-based XNA classifiers used for final residue-based prediction for proteins classified as XNA-binding. The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

As can be seen in Figure 18 and Table 8 the residue-based predictor for both polynucleotides (XNA) achieves a precision of 0.54 and a recall of 0.52 on average over the five folds using the thresholds determined on the cross-training data. To decide on the informative value of this predictor, it can be opposed to a random classifier. The precision of this random classifier is equal to the proportion of positive samples to all samples in the data, thus the calculation is the same as for the precision of the trained classifier. For the XNA data, this proportion is 0.17, which means in this setting, the performance of the trained predictor is more than three fold higher. As shown in Figure 18, the precision can even be raised to more than 0.8 at the cost of a decreased recall (0.2), which is a good attempt for determining binding regions in the protein.

DNA

	Threshold	P_+	R_+	FPR	P_-	R_-	Q2	MCC
Fold 1	0.05	0.61	0.47	0.06	0.90	0.94	0.87	0.46
Fold 2	-0.33	0.46	0.57	0.12	0.92	0.88	0.84	0.41
Fold 3	0.11	0.55	0.51	0.06	0.93	0.94	0.88	0.46
Fold 4	-0.36	0.49	0.52	0.08	0.93	0.92	0.87	0.43
Fold 5	0.07	0.53	0.58	0.10	0.91	0.90	0.84	0.46
Average		**0.53**	**0.53**	**0.08**	**0.92**	**0.92**	**0.86**	**0.44**

Table 9: The precision and recall values, the Q2 score, the FPR and the MCC of the residue-based DNA classifiers used for the final residue-based prediction for proteins classified as DNA-binding. The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

The performance of the DNA residue-based predictor is similar to the XNA one, as shown in Table 9 with an average precision of 0.53 and recall of 0.53. Because the proportion of binding residues to non-binding residues is smaller in this data set, a random predictor would only achieve a precision of 0.15. The curves in Figure 18 reveal that more accurate predictions at a level of 0.2 recall can reach 0.8 in terms of precision.

RNA

	Threshold	P_+	R_+	FPR	P_-	R_-	Q2	MCC
Fold 1	-0.29	0.70	0.43	0.07	0.82	0.93	0.80	0.43
Fold 2	-0.05	0.43	0.41	0.10	0.89	0.90	0.82	0.31
Fold 3	-0.40	0.63	0.52	0.08	0.89	0.92	0.84	0.48
Fold 4	-0.35	0.48	0.46	0.10	0.90	0.90	0.83	0.37
Fold 5	0.42	0.42	0.66	0.23	0.90	0.77	0.75	0.37
Average		**0.53**	**0.50**	**0.12**	**0.88**	**0.88**	**0.81**	**0.39**

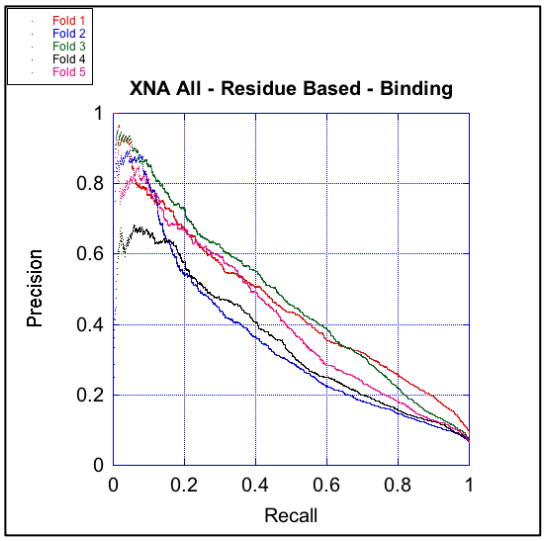
Table 10: The precision and recall values, the Q2 score, the FPR and the MCC of the residue-based RNA classifiers used for final residue-based prediction for proteins classified as RNA-binding. The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

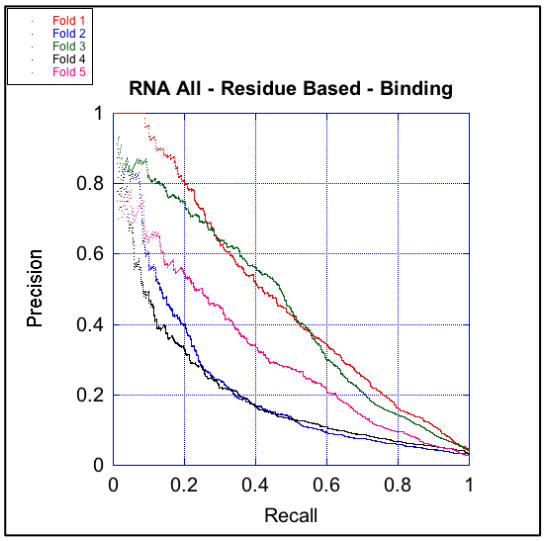
Also for the residue-based RNA data, the performance is similar with a precision of 0.53 and a recall of 0.50 (Table 10), while a random predictor for this data can

only achieve a precision of 0.2. By visual inspection of Figure 18 it stands out, that the performance variance among the folds is the highest for the RNA proteins and the lowest for the DNA proteins. The medium variance in the XNA data is a result of combining those two data sets. One explanation for the high variance case is that the RNA set contains about 40% less proteins than the DNA set.

Residue-Based Classifiers for Protein Score Calculation

To make the step from residue-based polynucleotide classification to the protein-based one, there is the necessity of specialized networks that have also experienced the signals from non-binding protein residues. Figure 19 illustrates the performance of the classifiers used to calculate the protein score as described in the methods section. The exact used thresholds, which were selected by evaluation on the cross-training data, can be seen in Table 11, Table 12 and Table 13. In contrast to the neural network models used for the final residue-bases predictions, these are trained and validated including the non-binding proteins to specialize them by providing the signals from those negative examples. Also the RNA-binding proteins were included in the DNA-binding training as negative examples and vice versa. Although they seem to be low in performance, i.e. none of the average precisions and recalls is above 0.45, they are good enough to provide a foundation for the final protein score, especially when comparing the performance to the fraction of binding residues in the particular datasets. Thus, the expected precision values for the random classifiers are 0.07 for XNA, 0.04 for DNA and 0.05 for RNA predictions calculated by the distribution of positive data samples among all residues in the particular datasets.





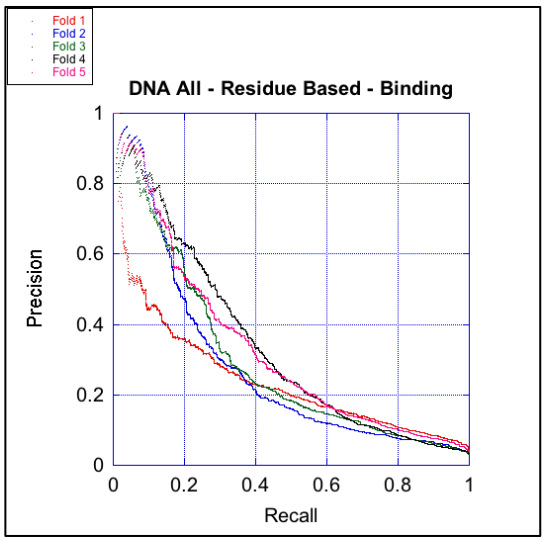


Figure 19: Precision plotted against the recall of the residue-based classifiers, which are trained including the non-binding proteins. Data is derived from the test parts of the 5-fold cross-validation and plotted for DNA, RNA and XNA.

	Threshold	P_+	R_+	FPR	P_-	R_-	Q2	MCC
Fold 1	0.09	0.52	0.38	0.03	0.94	0.97	0.92	0.40
Fold 2	-0.01	0.30	0.49	0.08	0.96	0.92	0.89	0.33
Fold 3	-0.56	0.54	0.41	0.03	0.96	0.97	0.94	0.44
Fold 4	0.13	0.42	0.39	0.04	0.96	0.96	0.93	0.36
Fold 5	0.08	0.40	0.48	0.05	0.96	0.95	0.92	0.40
Average		**0.44**	**0.43**	**0.05**	**0.96**	**0.95**	**0.92**	**0.39**

Table 11: The precision and recall values, the Q2 score, the FPR and the MCC of the residue-based XNA classifiers used for the calculation of the protein binding score. The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

	Threshold	P_+	R_+	FPR	P_-	R_-	Q2	MCC
Fold 1	-0.71	0.33	0.25	0.02	0.97	0.98	0.95	0.26
Fold 2	-0.70	0.28	0.33	0.03	0.98	0.97	0.95	0.28
Fold 3	-0.63	0.35	0.29	0.02	0.98	0.98	0.96	0.30
Fold 4	0.04	0.33	0.41	0.03	0.98	0.97	0.95	0.34
Fold 5	0.03	0.36	0.37	0.02	0.98	0.98	0.95	0.34
Average		**0.33**	**0.33**	**0.02**	**0.98**	**0.98**	**0.95**	**0.30**

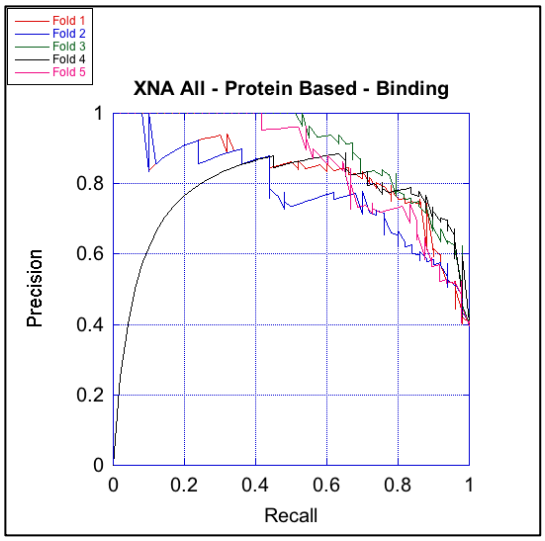
Table 12: The precision and recall values, the Q2 score, the FPR and the MCC of the residue-based DNA classifiers used for the calculation of the protein binding score. The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

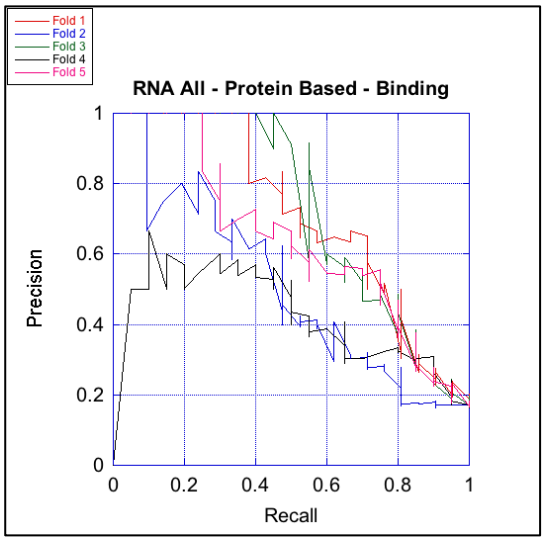
	Threshold	P_+	R_+	FPR	P_-	R_-	Q2	MCC
Fold 1	-0.53	0.48	0.45	0.02	0.98	0.98	0.95	0.44
Fold 2	-0.74	0.19	0.36	0.05	0.98	0.95	0.93	0.23
Fold 3	0.06	0.49	0.47	0.02	0.98	0.98	0.96	0.46
Fold 4	0.06	0.24	0.29	0.03	0.98	0.97	0.95	0.24
Fold 5	0.13	0.46	0.28	0.01	0.98	0.99	0.97	0.34
Average		**0.37**	**0.37**	**0.03**	**0.98**	**0.97**	**0.95**	**0.34**

Table 13: The precision and recall values, the Q2 score, the FPR and the MCC of the residue-based XNA classifiers used for the calculation of the protein binding score. The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

Protein-Based Prediction

Although the main benefit of the method is to distinguish between the different types of polynucleotides bound, this separation is based on single predictors for DNA, RNA and XNA. The following data presents their performance. Due to the small amount of data, the precision/recall curves show great variability in the beginning. However, they stabilizes as the recall gets higher.





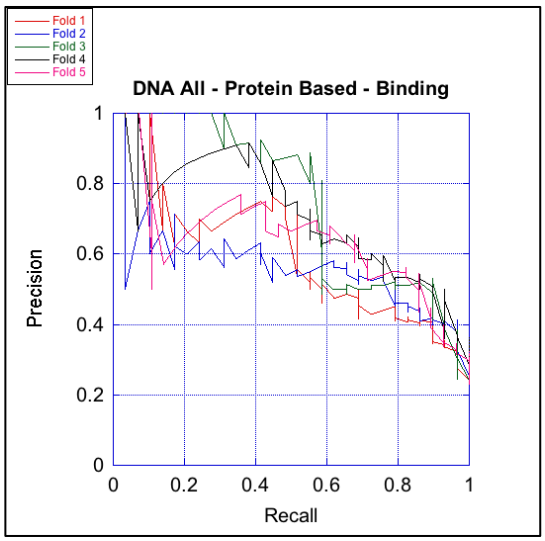


Figure 20: Precision plotted against the recall of the protein-based neural network models for XNA, DNA and RNA. The large steps mainly in the low recall area result from the low amount of data per fold. The data includes the particular binding proteins, as well as the non-binding ones.

XNA

	Threshold	P_+	R_+	FPR	P_-	R_-	Q2	MCC
Fold 1	50.00	0.79	0.76	0.14	0.84	0.86	0.82	0.63
Fold 2	39.00	0.77	0.68	0.14	0.80	0.86	0.79	0.56
Fold 3	40.00	0.85	0.69	0.08	0.82	0.92	0.83	0.64
Fold 4	45.00	0.80	0.71	0.12	0.82	0.88	0.81	0.60
Fold 5	35.00	0.71	0.83	0.22	0.88	0.78	0.80	0.60
Average		**0.78**	**0.73**	**0.14**	**0.83**	**0.86**	**0.81**	**0.61**

Table 14: Precision and recalls, the Q2 score, the FPR and the MCC of the protein-based XNA predictors at the given threshold. The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

The best working single predictor regarding the performance measured as precision and recall is the combined predictor for DNA and RNA (XNA) as can be seen in Table 14 or Figure 20 respectively. Achieving a very high precision of 0.78 and a descent recall of 0.73 it shows a high reliability of its predictions. A random classifier would achieve a precision of only 0.35 according to the used data.

DNA

	Threshold	**P_+**	**R_+**	**FPR**	**P_-**	**R_-**	**Q2**	**MCC**
Fold 1	55.00	0.75	0.41	0.04	0.84	0.96	0.83	0.47
Fold 2	45.00	0.52	0.76	0.21	0.91	0.79	0.78	0.49
Fold 3	40.00	0.68	0.59	0.09	0.88	0.91	0.84	0.53
Fold 4	47.00	0.63	0.66	0.12	0.89	0.88	0.83	0.53
Fold 5	39.00	0.55	0.79	0.19	0.93	0.81	0.80	0.53
Average		**0.63**	**0.64**	**0.13**	**0.89**	**0.87**	**0.82**	**0.51**

Table 15: Precision and recalls, the Q2 score, the FPR and the MCC of the protein-based DNA predictors at the given threshold. The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

According to Table 15 precision and recall of the DNA protein-based classifier are 0.63 and 0.64. As the used dataset also includes RNA binding proteins, this may be the reason for the low performance compared to the XNA predictor. This problem will be solved later by applying the algorithm to distinguish between those polynucleotides.

RNA

	Threshold	P_+	R_+	FPR	P_-	R_-	Q2	MCC
Fold 1	57.00	0.67	0.67	0.07	0.93	0.93	0.89	0.60
Fold 2	40.00	0.58	0.33	0.05	0.87	0.95	0.85	0.36
Fold 3	35.00	0.54	0.65	0.11	0.93	0.89	0.85	0.50
Fold 4	50.00	0.42	0.50	0.14	0.90	0.86	0.80	0.34
Fold 5	40.00	0.86	0.30	0.01	0.88	0.99	0.88	0.46
Average		**0.61**	**0.49**	**0.08**	**0.90**	**0.92**	**0.85**	**0.45**

Table 16: Precision and recalls, the Q2 score, the FPR and the MCC of the protein-based RNA predictors at the given threshold. The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

The RNA binding predictor has a similar performance to the DNA one, but with slightly better recall, as can be seen in Table 16. Compared to a random predictor with the expected precision of 0.14 it achieves descent performance though. Like the DNA predictor, the neural network model may have its problems since the data contains the other type of protein-polynucleotide complexes, namely protein-DNA, as well.

Polynucleotide Type Prediction

	XNA				DNA				RNA			
	P_+	R_+	P_-	R_-	P_+	R_+	P_-	R_-	P_+	R_+	P_-	R_-
Fold 1	0.79	0.76	0.84	0.86	0.91	0.34	0.83	0.99	0.67	0.67	0.93	0.93
Fold 2	0.77	0.68	0.80	0.86	0.61	0.66	0.89	0.87	0.67	0.19	0.85	0.98
Fold 3	0.83	0.69	0.81	0.90	0.80	0.55	0.87	0.96	0.80	0.60	0.93	0.97
Fold 4	0.80	0.71	0.82	0.88	0.81	0.59	0.88	0.96	0.47	0.45	0.89	0.90
Fold 5	0.68	0.85	0.89	0.74	0.58	0.79	0.93	0.83	1.00	0.15	0.86	1.00
Avg.	**0.77**	**0.74**	**0.83**	**0.85**	**0.74**	**0.59**	**0.88**	**0.92**	**0.72**	**0.41**	**0.89**	**0.96**

Table 17: Precision and recall values for the final predictor, which distinguishes between the type of binding (XNA, DNA, RNA or none). The values are given for each fold/test-part of the 5-fold cross-validation as well as an average over those folds.

The real novelty of the method is the separation of proteins that bind either DNA, RNA or none of both. As can be seen in Table 17, the precisions for the DNA and RNA binding prediction are at 0.74 and 0.72 respectively. In comparison to the single classifiers, this means a striking increase of accuracy, with just a little loss

in recall. The XNA prediction performance of the combined predictor is the same as the single classifier, due to the functioning of the method, meaning the combined classifier just adds additional restrictions to the DNA and RNA binding prediction.

.

Conclusion and Ideas

With SomeNA, a novel type of binding prediction method becomes available for scientific use. Its highly precise protein-based predictions combined with the possibility to investigate polynucleotide-binding regions give new opportunities to detect polynucleotide binding proteins like transcription factors from sequence data only. This tool will also be helpful for genome annotation.

In the following improvements, which could increase the performance of the predictor further, are detailed. At the end of this chapter, the current features of SomeNA are summarized.

New features like a global encoding of the sequence for the protein-based predictor could provide a signal that is missing when using only the amino acid composition. An idea how to solve this problem would be to compute a feature out of all possible combinations of residue triplets.

Less restrictive redundancy reduction could solve the problem of rare data. A HVAL of zero is very restrictive and based on the assumption that beyond this similarity threshold, a prediction from homology can achieve better results. Nevertheless, more data could provide stronger signals to the machine-learning algorithm. The disadvantage of this approach is less comparability because the training and test cases would differ less from each other.

More Data could also be achieved by not restricting the used PDB identifiers to those incorporated in PDIDB or PRIDB.

Models using different distance thresholds, and a clever combination of the predictors trained on those different distances could improve the reliability of the residue-based predictions, which directly influences the protein-based predictions.

Other machine-learning algorithms like support vector machines that use kernels might be worth testing, although my personal preference goes towards

the neural networks due to their fast prediction speed and ease of use as they need no kernel.

The use of direct multiclass predictions, which can be done by neural networks. This means that a multilayer perceptron can have an arbitrary number of output nodes, which would be a node for DNA, one for RNA and another for non-binding in the case of this work. Other machine-learning algorithms rely on pairwise evaluation of the predictors to solve multiclass classification tasks.

Although these ideas could lead to an improvement of the method, SomeNA is already a good classificator with the ability to bring new insights to its users. With the possibility of distinguishing between the two types of polynucleotide-binding proteins, namely DNA and RNA, scientists now have an additional tool to investigate whole genomes based on the complex relations between DNA- and RNA-binding proteins. These relations can be observed for example in transcription factors. Another benefit of the method is the included prediction of binding-residues. This provides an easy way to find areas in the protein involved in polynucleotide binding, which can for its part serve to identify the corresponding sequences on the polynucleotides themselves.

Annex

Additional Figures

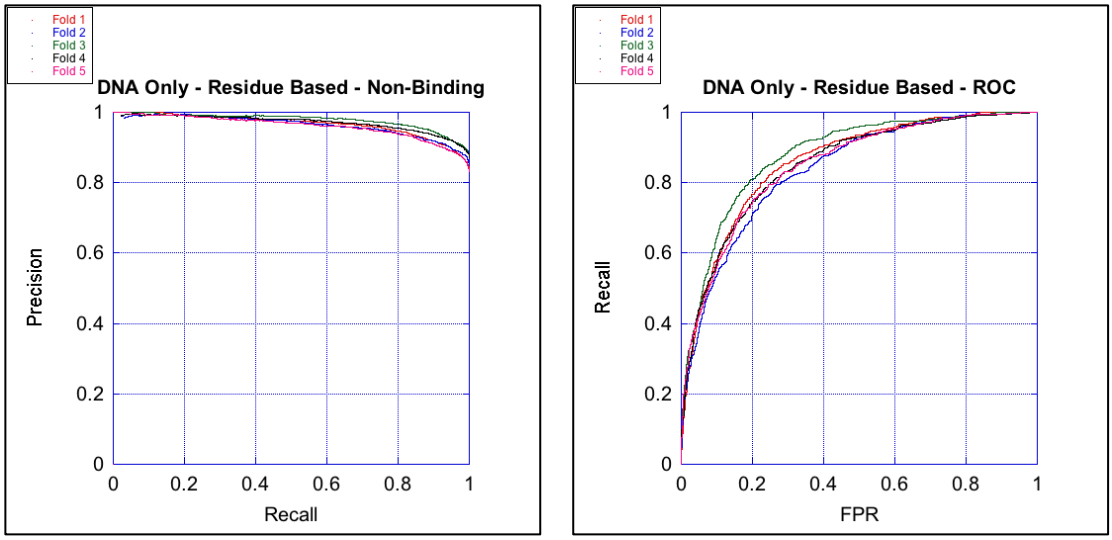


Figure 21

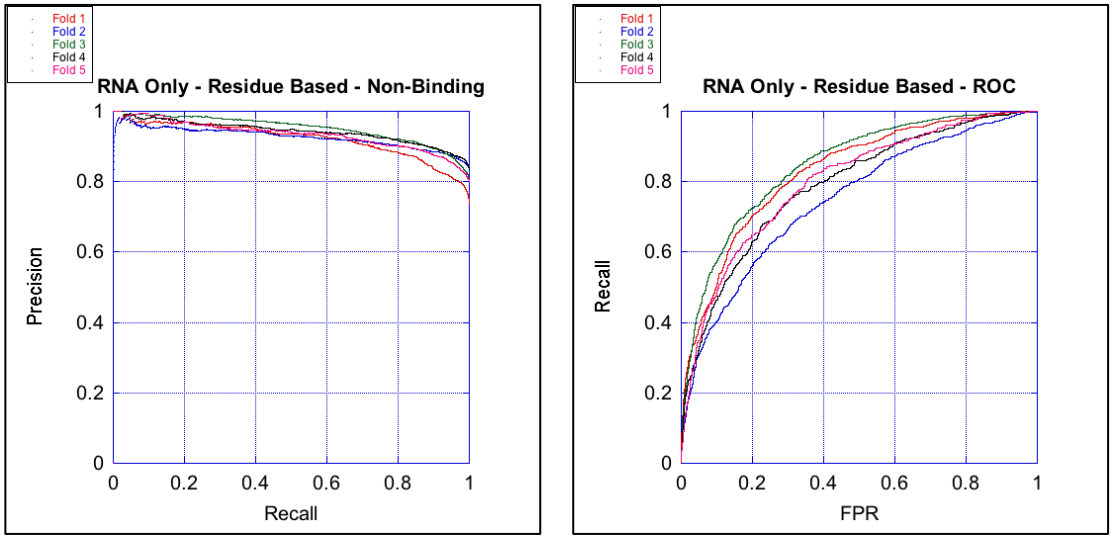
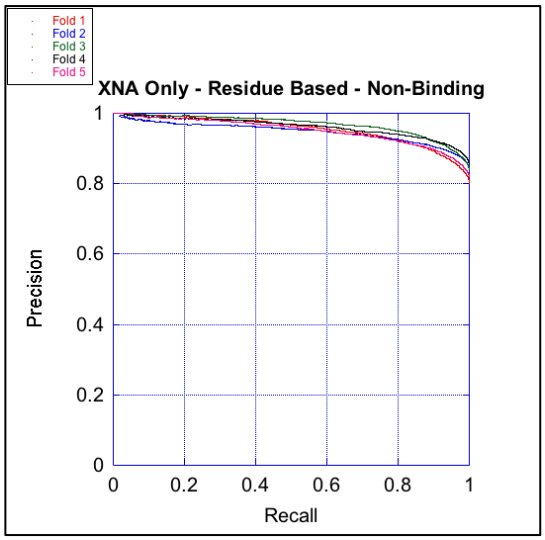


Figure 22



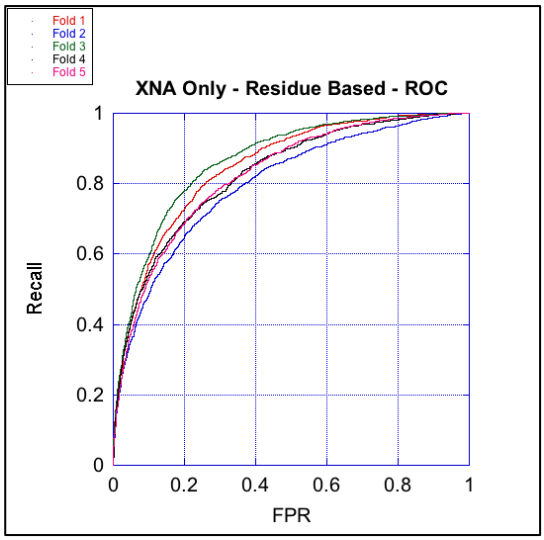
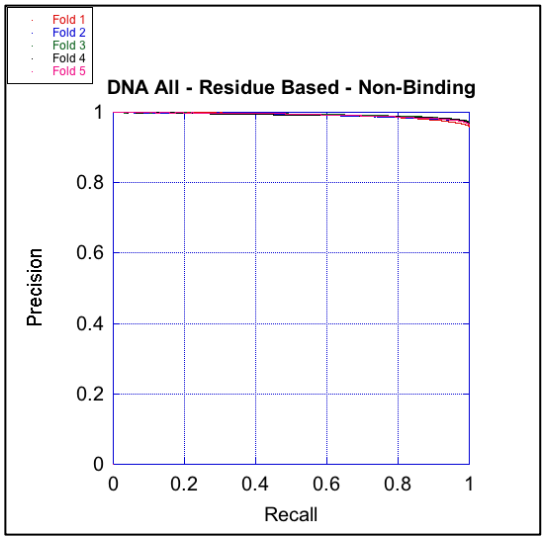


Figure 23



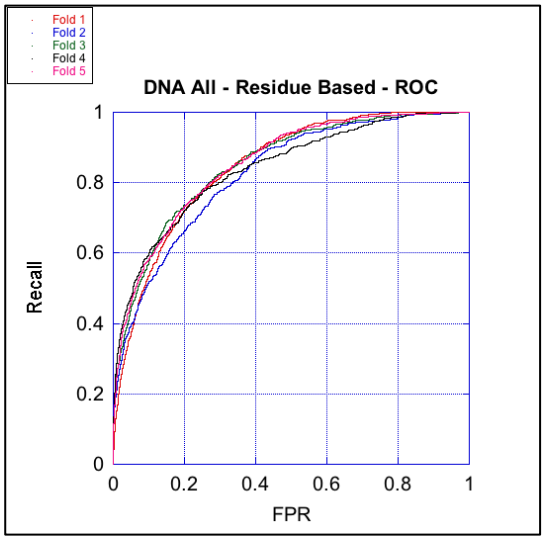
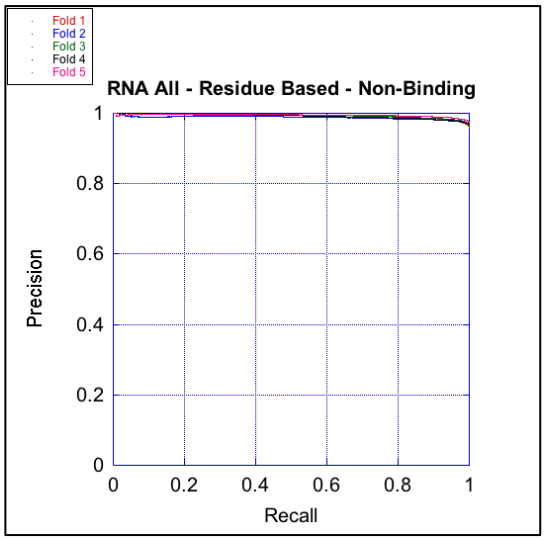


Figure 24



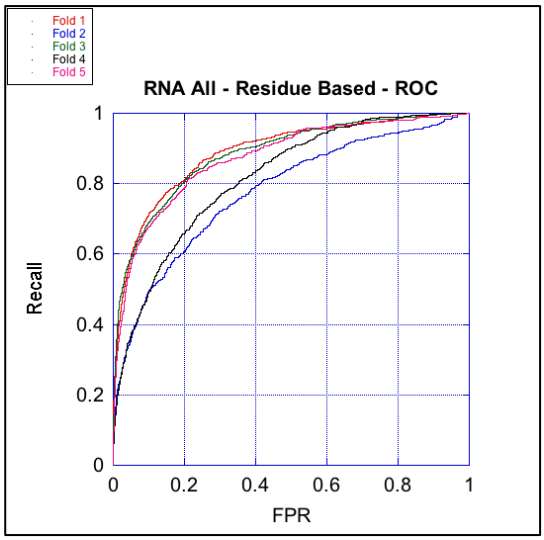
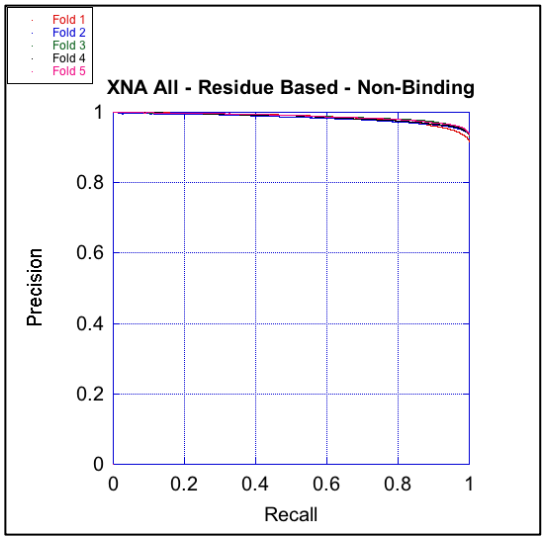


Figure 25



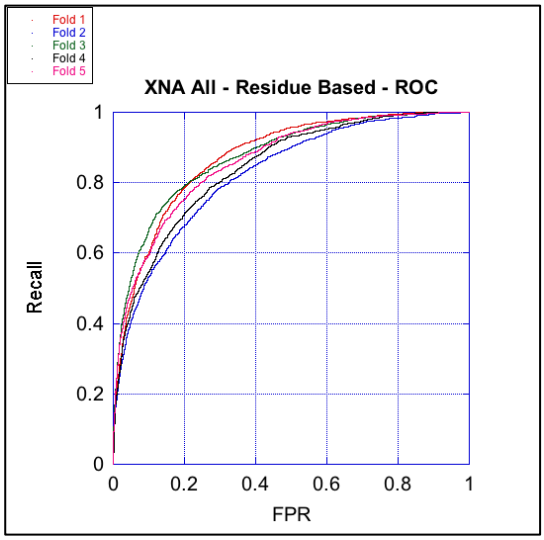
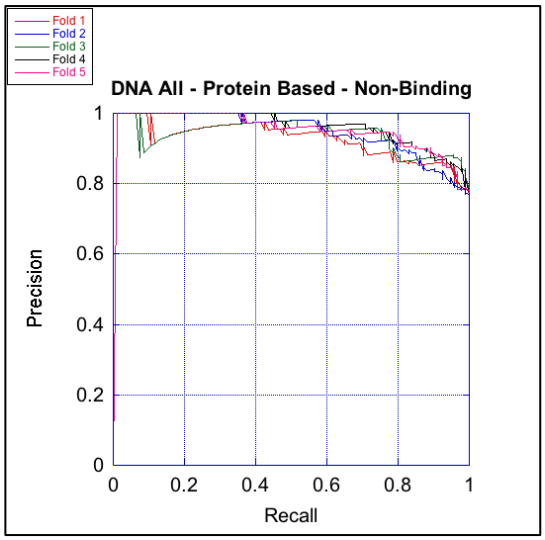


Figure 26



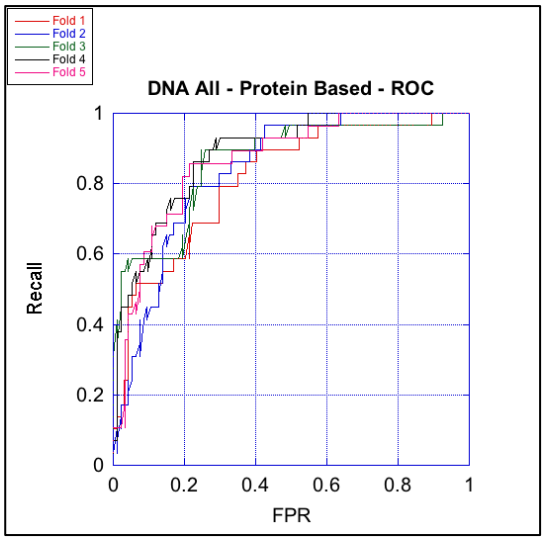
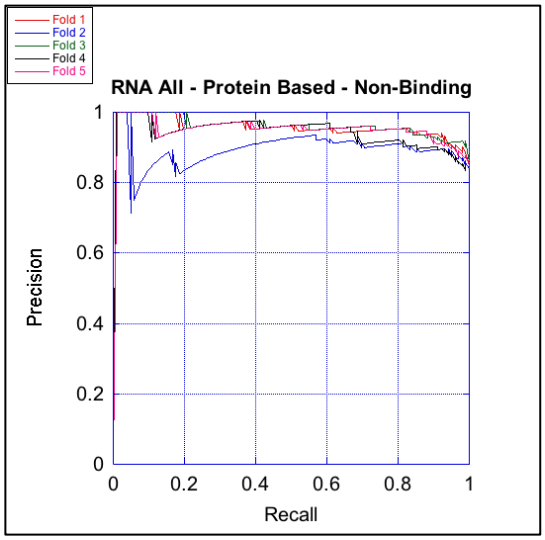


Figure 27



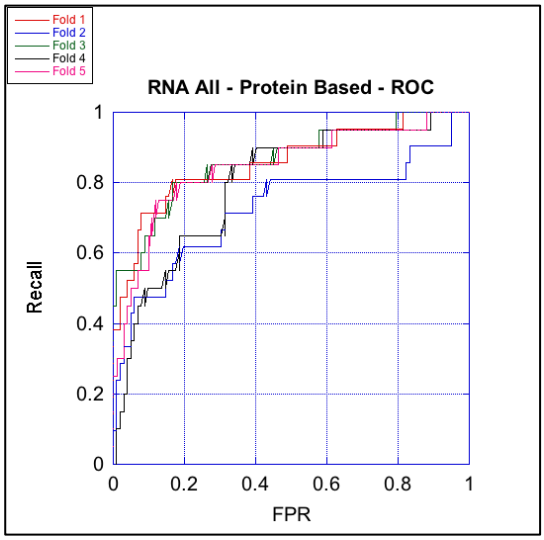


Figure 28

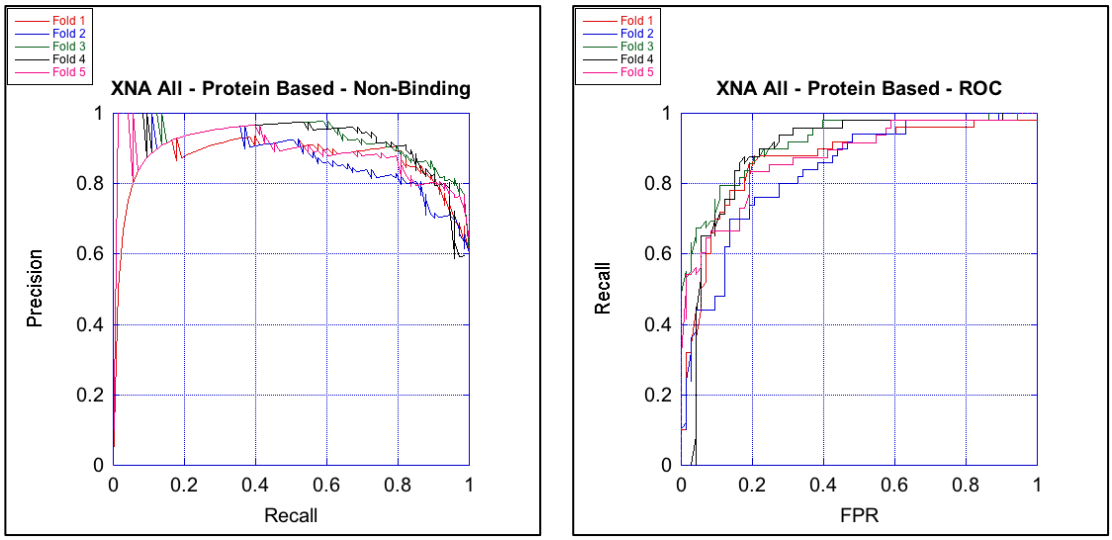

Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
XNA All - Protein Based - Non-Binding
Precision
Recall
0
0.2
0.4
0.6
0.8
1
XNA All - Protein Based - ROC
Recall
FPR

Figure 29

Bibliography

A. Lupas, M. Van Dyke, J. Stock. "Predicting coiled coils from protein sequences." *Science* 252 (1991): 1162–1164.

A. Schlessinger, B. Rost. "Protein flexibility and rigidity predicted from sequence." *Proteins* 1, no. 61(1) (2005): 115-126.

Avner Schlessinger, Marco Punta, Guy Yachdav, Laszlo Kajan, Burkhard Rost. "Improved Disorder Prediction by Combination of Orthogonal Approaches." *PLoS ONE* 4, no. 2 (2009): e4433.

B. Rost, G. Yachdav, J. Liu. "The PredictProtein Server." *Nucleic Acids Research* 32, no. Web Server (2003): W321-W326.

Bruce Alberts, Alexander Johnson, Julian Lewis, Martin Raff, Keith Roberts, Peter Walter. "Molecular Biology of the Cell, 4th edition." 2002.

Changhui Yan, Michael Terribilini, Feihong Wu, Robert L Jernigan, Drena Dobbs,Vasant Honavar. "Predicting DNA-binding sites of proteins from amino acid sequence." *BMC Bioinformatics* 7, no. 262 (2006).

Consortium, The Gene Ontology. "Gene ontology: tool for the unification of biology." *Nature Genetics* 25, no. 1 (2000): 25-29.

Consortium, The UniProt. "Reorganizing the protein space at the Universal Protein Resource (UniProt)." *Nucleic Acids Research* 40, no. D1 (2011): 71-75.

Daniel Nathans, Hamilton O. Smith. "Restriction Endonucleases in the Analysis and Restructuring of DNA Molecules." *Annual Review of Biochemistry* 44 (1975): 273-293.

E. Ferrada, F. Melo. "Effective knowledge-based potentials." *Protein Science* 18, no. 7 (July 2009): 1469-1485.

E.E. Pryor, E.A. Waligora, B. Xu, S. Dellos-Nolan, D.J. Wozniak, T. Hollis. "The Transcription Factor AmrZ Utilizes Multiple DNA Binding Modes to Recognize Activator and Repressor Sequences of Pseudomonas aeruginosa Virulence Genes." *Plos Pathogenic* 8 (2012): e1002648-e1002648.

E.Y.D. Chua, D. Vasudevan, G.E. Davey, B. Wu, C.A. Davey. "The mechanics behind DNA sequence-dependent properties of the nucleosome." *Nucleic Acids Research* 40, no. 13 (2012): 6338-6352.

García, Salvador Fandiño. *AI::FANN.* 2009. http://search.cpan.org/~salva/AI-FANN-0.10/lib/AI/FANN.pm (accessed 2012).

Gilbert, W. "Origin of life: the RNA world." *Nature* 319, no. 6055 (1986): 618.

Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov and Philip E. Bourne. "The Protein Data Bank." *Nucleic Acids Research* 28, no. 1 (2000): 235-242.

Manish Kumar, Michael M. Gromiha, Gajendra PS Raghava. "Identification of DNA-binding proteins using support vector machines and evolutionary profiles." *BMC Bioinformatics* 8, no. 463 (2007).

Michael Remmert, Andreas Biegert, Andreas Hauser, Johannes Söding. "HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment." *Nature Methods* 9 (2012): 173-175.

Nissen, Steffen. *Fast Artificial Neural Network Library (FANN).* 2012. http://leenissen.dk/fann/wp/ (accessed 2012).

R.R. Walia, C. Caragea, B.A. Lewis, F. Towfic, M. Terribilini, Y. El-Manzalawy, D. Dobbs, V. Honavar. "Protein-RNA Interface Residue Prediction using Machine Learning: An Assessment of the State of the Art." *BMC Bioinformatics* 13, no. 98 (2012).

RK Saiki, S Scharf, F Faloona, KB Mullis, GT Horn, HA Erlich, N Arnheim. "Enzymatic amplification of beta-globin genomic sequences and restriction site analysis for diagnosis of sickle cell anemia." *Science* 230, no. 4732 (1985): 1350-1354.

S. Mika, B. Rost. "UniqueProt: Creating representative protein sequence sets." *Nucleic Acids Research* 31, no. 13 (2003): 3789-3791.

Shen Wei, Christian Schäfer, Burkhard Rost. "Extracting binding residues from the Protein Data Bank." 2012.

Stephen F. Altschul, Thomas L. Madden, Alejandro A. Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, David J. Lipman. "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs." *Nucleic Acids Research* 25 (1997): 3389-3402.

T. Norambuena, F. Melo. "The Protein-DNA Interface database." *BMC Bioinformatics* 11, no. 262 (2010).

The Comprehensive Perl Archive Network. 2012. http://www.cpan.org/ (accessed 2012).

Yanay Ofran, Burkhard Rost. "ISIS: interaction sites identified from sequence." *Bioinformatics* 23, no. 2 (2007): e13-e16.

Yao Chi Chen, Jon D. Wright, Carmay Lim. "DR_bind: a web server for predicting DNA-binding residues from the protein structure based on electrostatics, evolution and geometry." *Nucleic Acids Research* 40, no. W1 (2012): W249-W256.