Probabilistic Models to Detect Important Sites in Proteins

Dissertation
zur Erlangung des mathematisch-naturwissenschaftlichen Doktorgrades
"Doctor rerum naturalium"
der Georg-August-Universität Göttingen

im Promotionsprogramm Computer Science (PCS)
der Georg-August University School of Science (GAUSS)

vorgelegt von

Truong Khanh Linh Dang
aus Tien Giang

Göttingen, 2021

Betreuungsausschuss

Prof. Dr. Stephan Waack,
Institut für Informatik, Georg-August-Universität Göttingen

Prof. Dr. Carsten Damm,
Institut für Informatik, Georg-August-Universität Göttingen

Mitglieder der Prüfungskommission

Referent:	Prof. Dr. Stephan Waack, Institut für Informatik, Georg-August-Universität Göttingen.
Korreferent:	Prof. Dr. Carsten Damm, Institut für Informatik, Georg-August-Universität Göttingen.

Weitere Mitglieder der Prüfungskommission

Prof. Dr. Michael Habeck,
University Hospital Jena.

Dr. Johannes Söding,
Quantitative and Computational Biology
Max Planck Institute for Biophysical Chemistry, Göttingen.

Prof. Dr. Tim Beißbarth,
Institut für medizinische Bioinformatik, Universitätsmedizin Göttingen.

Prof. Dr. Marcus Baum,
Institut für Informatik, Georg-August-Universität Göttingen.

Tag der mündlichen Prüfung
24. September 2020

Abstract

Proteins are molecular machines playing almost every fundamental role in activities of life. Their biological functions are mostly driven through conformational transitions and interaction interfaces with other bio-molecules such as DNA sequences, proteins and other ligands. In quest of the mechanism underlying protein functions, I conducted two projects aiming, firstly, to explore the structural change of proteins via identifying their rigid bodies, and secondly, to devise new sequence-based features to predict DNA-binding sites in proteins.

Despite many previous efforts to calculate rigid domains in proteins, it is still highly desirable to develop new segmentation algorithms which are able to efficiently segment high-throughput of proteins, meanwhile to avoid protein-dependent parameters tuning such as the number of rigid domains. Thus, I introduce a new rigid domain segmentation method where I use a graph whose vertices are amino acids to represent multiple conformational states of a protein. This graph is later reduced by a coarse graining such as the Louvain clustering algorithm. Afterward, the domain-wise relationships among clusters in the reduced graph were inferred through a binary labeling of its edges which becomes feasible thanks to the line graph transformation and generalized Viterbi algorithm. Because of the binary labeling, our method does not require the number of rigid domains as an input parameter like other existing methods. I validate our graph-based method on 487 examples from DynDom database and compare our segments with other methods on several proteins whose structural changes range from medium to large and their molecular motions have been studied extensively in the literature. The algorithm code as well as usage instruction is available at https://github.com/dtklinh/GBRDE.

In the second project, the identification of DNA-binding sites in proteins could be obtained either through structure- or sequence-based approaches. In spite of obtaining good results, structure-based methods require protein 3D structures which are expensive and time-consuming. In contrast, the sequence-based ones are efficiently applicable to entire protein databases, yet demand carefully designed features. Thus, I present a new information theoretic feature extracted from the Jensen–Shannon Divergence (JSD) where I harvest the differences between amino acids distributions of binding and non-binding sites. For the evaluation, I ran a five-fold cross validation on 263 proteins with Random Forest (RF) classifier along with features comprising of our new sequence-based feature and several popular ones such as position-specific scoring matrix (PSSM), orthogonal binary vector (OBV), and secondary structure (SS). The results show that by concatenating our features, there is a sig-

nificant improvement of RF classifier performance in terms of sensitivity and Matthews correlation coefficient (MCC).

Acknowledgements

Hereby, I wish to express my sincere gratitude towards all people whose assistance was a milestone in the completion of my projects. First of all, I would like to pay my special regards to my supervisor Prof. Dr. Stephan Waack who continuously support my PhD study. His motivation, immense knowledge, and guidance helped me all the time of research and writing of this thesis, especially during the pandemic crisis.

Additionally, many thanks goes to my second supervisor Prof. Dr. Carsten Damm for his patience and valuable feedback during my project presentations. Furthermore, I would like to thank other members in my examination board Dr. Johannes Söding, Prof. Dr. Michael Habeck, Prof. Dr. Tim Beißbarth, and Prof. Dr. Marcus Baum for investing their precious time.

Moreover, I dedicate many thanks to my current and former coworkers at our institute. In particular, I am grateful to Dr. Mehmet Gültas for enlightening me the first glance of research. Even though he moved to another faculty, his office door was always open whenever I needed his advice. I also thank Dr. Daniel Honsel for proofreading the thesis, and Gunnar Krull for his technical support.

Also, I wish to show my gratitude to several collaborators who were very supportive during my PhD time. I thank Dr. Johannes Söding for dedicating his time and energy during our discuss, and consequently resulted in new ideas and inspiration. Furthermore, I am greatly grateful towards Prof. Dr. Michael Habeck and his colleague Dr. Thach Nguyen for our collaboration. During project discussions, they gave me their insights and other useful information as to conduct the research properly.

My special thanks also goes to my friends from Göttingen and other cities around Germany. I am thankful to have Isolde as my flatmate, who is always available whenever I need her help. In particular I am indebted to Isolde for her willingness to teach me German, and to carefully review and comment my thesis. I also thank Sigrid and Rüdiger for their helpfulness and immense knowledge about Göttingen. Hiking, cycling, and talking with them were always an enjoyable time to keep me away from stress during my PhD. Additionally, I am grateful to have Nguyen family from Brigachtal as my second Vietnamese family in Germany. The family did not only offer me a mountain PhD retreat, but they also cooked delicious Vietnamese foods which helped me feel at home. Their encouragements and constructive feedback were invaluable impacts on my completion of the PhD study. Besides, I would like to deeply express my special gratitude towards Dr. To Vinh for his priceless

support and mentorship. His energetic motivational speeches and tremendous knowledge were always illuminating our conversation, thus resulted in inspiration and new ideas for my PhD study.

Finally, my deepest gratitude goes to my parents and my sister who were unconditionally supporting my education. Undoubtedly, I could not have accomplished this work without their support and motivation. With all my sincerity, I dedicate this dissertation to them.

Contents

1. Introduction **1**
- 1.1. Research Questions . . . 4
- 1.2. Scope of the Thesis . . . 4
- 1.3. Impact . . . 5
- 1.4. Structure of the Thesis . . . 6

2. Biological Backgrounds **7**
- 2.1. Proteins . . . 7
- 2.2. Conformational Change in Proteins . . . 8
- 2.3. Features Extraction for DNA-binding Sites in Proteins Prediction . . . 8
  - 2.3.1. Position-specific Scoring Matrix . . . 9
  - 2.3.2. Secondary Structure . . . 9
  - 2.3.3. Orthogonal Binary Vector . . . 9
- 2.4. Biological Information Sources . . . 10
  - 2.4.1. HH-suite . . . 10
  - 2.4.2. Protein Data Bank . . . 10
  - 2.4.3. Non-redundant DynDom Database . . . 10
  - 2.4.4. TM-Score . . . 11

3. Mathematical Backgrounds **13**
- 3.1. Information Theory . . . 13
  - 3.1.1. Shannon Entropy . . . 13
  - 3.1.2. Conditional Entropy . . . 14
  - 3.1.3. Mutual Information . . . 14
  - 3.1.4. Kullback–Leibler Divergence . . . 14
  - 3.1.5. Jensen–Shannon Divergence . . . 14
- 3.2. Machine Learning Techniques . . . 15
  - 3.2.1. Random Forest . . . 15
  - 3.2.2. Conditional Random Field . . . 17
- 3.3. The Generalized Viterbi Algorithm . . . 19
  - 3.3.1. Pseudo Algorithm . . . 20
- 3.4. Constant Potts Model in Louvain Algorithm . . . 24
- 3.5. Outliers Detection . . . 26

4. Probabilistic Models to Detect Important Sites in Proteins **27**
4.1. Identification of Rigid Domains in Proteins 27
4.1.1. Protein Graph Constructions 27
4.1.2. Coarse Graining of the Protein Graph 29
4.1.3. Line Graph Transformation 30
4.1.4. Applying the Outlier Detection 30
4.1.5. Applying the Generalized Viterbi Algorithm to the Labels Inference 31
4.1.6. Pseudo Code 33
4.1.7. Finalizing the rigid domain segmentation 34
4.2. Identification of DNA-Binding Sites in Proteins 34
4.2.1. Materials 34
4.2.2. Methods 35

5. Results **39**
5.1. Identification of Rigid Domains in Proteins using Graph-based Model . . . 39
5.1.1. Rigid domains of Adenylate Kinase 39
5.1.2. Rigid Segmentation Benchmark 41
5.1.3. Rigid segmentation on various structural transitions 46
5.2. Identification of DNA-Binding Sites in Proteins Using JSD 49
5.2.1. Cross Validation on Benchmark 49
5.2.2. Position Analysis of the MYC-MAX Protein 51

6. Discussion **55**
6.1. Discussion: Rigid Domains in Proteins Detection 55
6.1.1. Coarse-graining Procedure 55
6.1.2. Line Graph Transformation 59
6.1.3. Running Time 61
6.1.4. Merging Algorithm 61
6.2. Discussion: Novel Sequence-based Feature Engineering 63

7. Conclusion **65**
7.1. Summary 65
7.2. Outlook 65

Bibliography **67**

List of Acronyms **76**

Glossary **77**

List of Figures **83**

List of Algorithms and Listings **87**

List of Tables **89**

A. Appendix **91**
A.1. The Calculation of Rigid Domains in Proteins 91
A.2. The Prediction of DNA-binding sites in Proteins 115
A.3. CRF-based Models for Protein-protein Interaction Site Predictions 129

1. Introduction

A protein is a macromolecule folded from a chain of amino acids and kept in shape through peptide bond, sulfhydryl bonds, hydrogen bonds and van der Waals forces [1]. Proteins play almost every essential role in molecular activities of life such as binding to antigens to protect the body (antibodies) [2]; catalyzing thousands of chemical reactions in cells (enzymes) [3, 4]; transmitting signals to orchestrate many processes among cells, tissues and organs (messenger proteins) [5, 6]; providing structure for cells and support forming connective tissues (fibrous proteins) [7, 8]; and helping to circulate other vital elements around the body (transport and/or storage proteins) [9].

Thus, with the essential roles of proteins in nature, the quest to understand how proteins work is the crucial key to decipher how lives function. One important step to have a deep insight of underlying molecular mechanism of proteins is to understand how proteins interact with DNA to maintain and transmit the genetic information. For example, RNA polymerase and other transcription factors proteins bind to a specific region of genome such as enhancer and promoter to either promote or inhibit a generation of certain proteins. This process of gene expression enables cells to only produce a certain type of proteins they need for their metabolism, and thus leads to cells differentiation. In order to have a better understanding underlying the DNA-protein binding mechanism, I investigate to the following aspects of proteins. Firstly, I calculate rigid domains in proteins which is the initial step to understand the underlying motion of proteins. Secondly, I use Conditional Random Field (CRF)-based machine learning technique to improve the prediction of protein-protein interfaces. This knowledge will assist us to have a better understanding of how proteins team up into a complex. Nonetheless, this project will not be included in this thesis, but could be found in our publication [10]. Finally, I propose a new sequence-based feature to improve the identification of DNA-binding sites in proteins.

Identification of rigid domains in proteins

For the first research question, I investigate how to calculate rigid domains in proteins. A protein function is often determined through its large-scale structural transitions [11]. One of a reasonable approach to grasp such transitions is to partition proteins into rigid domains from their structures of various states. Consequently, one could analyze protein movements through hinge and shear motions (see Section 2.2) of those domains [12].

With a growing number of experimental protein structures, there is a need to develop a method which can automatically identify conserved domains in proteins without tedious parameters tuning. One can use such software, for instance, to study the trajectories of a protein motion at the level of rigid domains and consequently obtains a certain insight of its large-scale dynamic as well as its significant sites located along interfaces among rigid domains [13].

There are a few computational methods developed to identify rigid domains in proteins. DynDom [14] calculates protein rigid bodies by clustering its corresponding rotation vectors. The key idea underneath this method is that a residue rotation vector which describes the displacement of the residue between two conformations can be represented as a "rotation-point" in a rotation space. In an ideal situation, all rotation-points associated with one rigid domain will collapse into a single point. However, due to the unavoidable noise, those rotation-points instead tend to be in close proximity in rotation space and could be grouped by a certain clustering algorithm. Another approach to segment proteins into rigid bodies is to detect hinge residues such as Hingefind [15]. Given a pair of protein structures, this algorithm partitions a protein into rigid domains by adaptively select a residue using the Kabsch least-square fitting [16]. Another algorithm is RigidFinder [17] which iteratively calculates rigid domains in a protein via a dynamic programming which optimizes the rigidity of the segments. The performance of this method is heavily dependent on the selection of cutoff parameters which are not obvious to be determined. All of the above methods do not support multiple input structural conformations but instead they support only two.

Several other approaches have been proposed to overcome the above shortcoming which imposes the limitation of only two input structures. Ponzoni et al. introduced Spectrus [18] where they applied a spectral clustering algorithm to distance fluctuations. The quality score is usually used to suggest the number of the domains, which sometimes gives ambiguous results. Habeck and Nguyen developed a probabilistic approach [19] where they model protein motion as a combination of rotations and translations. They modeled protein rigid domains and their motions via parameters which were inferred by the Bayes approach [20] and the Gibbs sampling method [21]. Even though those above methods support multiple input structures, they require users to set up the initial parameters as well as the number of rigid domains. These methods search in their parameter space only locally and thus are probably trapped in local optimal points. Consequently, they require many restarts with various initial parameters set to work with.

Other methods using molecular dynamic simulation or an elastic network model enable users to predict rigid domains from a single structure. HingeProt [22] and Domain Finder [23] use elastic network models to predict hinge residues by analyzing the relationship between two slowest frequency normal modes which represent the global movements of large domains. However, when a conformational change involves multiple modes, it is, in

general, not clear how strongly those modes contribute to the movement. FlexOracle [24] predicts hinge residues by searching split points with minimal energetic impact.

In spite of the rich literature for the detection of rigid domains in proteins, there is still a need to develop algorithms which are robust, reliable, and able to handle high-throughput data and do not require parameters tuning in the same time.

Identification of DNA-binding sites in proteins

For the second research question, I investigate the possibility to further extract a sequence-based feature which is helpful for the prediction of DNA-binding sites in proteins. Transcription factors, the proteins which are able to bind to DNA, play essential roles to undertake several biological functions of life such as transcription, translation and gene regulation [25]. Thus, the identification of DNA-binding sites in proteins opens a new perspective to explore the underlying molecular mechanisms of these interactions. These binding residues in proteins enable us to understand how proteins work, thus consequently help for new drugs discovery [26, 27].

The methods developed for detection of DNA-binding sites in proteins are either structure-based or sequence-based approaches. Though structure-based methods usually obtain promising results, they require protein structures determined through experiments which are costly and time-consuming. To overcome the burdens of experimental approaches, it is desirable to develop numerical approaches which are able to handle high-throughput data while maintaining proper performances. A widely used computational approach is to construct a profile for each residue and to determine their DNA-binding properties via a classifier. In order to create an appropriate profile, many sequence-based features have been created such as amino acid frequency, position-specific scoring matrix (PSSM), BLOSUM62 matrix, sequence conservation [26, 28, 29, 30, 31, 32]. For instance, Westhof et al. developed RBscore [1], a support vector machine (SVM) approach to identify DNA-binding sites in proteins using physicochemical and evolutionary features along with a residue neighboring network [33]. BindN [29] developed by Wang et al. utilized sequence-based features derived from the side chain pKa value, the hydrophobicity index and molecular mass of an amino acid. BindN+ [34], an upgraded version of BindN, additionally added evolutionary information such as PSSM into feature space to improve the performance. DISIS [30] is another method for detection of DNA-binding sites in proteins which additionally utilized predicted structural features such as secondary structure (SS), solvent accessibility and globularity. DP-Bind [32] detects DNA-binding sites through PSSM empowered by various classifiers such as SVM, kernel logistic regression and penalized logistic regression. Many machine learning techniques have been used in order to predict DNA-binding sites in proteins. Besides SVMs, researchers have deployed other

[1] http://ahsoka.u-strasbg.fr/rbscore/

methods such as neural network [35, 36], naive Bayes classifier [31], and Random Forest classifiers [26, 37, 38].

1.1. Research Questions

In this thesis, I present the results of several projects aiming to explore the mechanism of proteins and their DNA-binding sites property. To pursue this aim, I focus on and investigate into the following research questions.

- **RQ 1:** Could we calculate protein rigid domains from different conformational states to infer protein transformation in large scale?

This question hence drives to the following more detailed subquestions, which I also investigate in this thesis:

- **RQ 1.1:** Could the number of rigid domains automatically be determined?
- **RQ 1.2:** What are the effects of the coarse graining and the line graph transformation?

Additionally, I consider a further main research question with a focus on the prediction of DNA-binding sites in proteins:

- **RQ 2:** Is it possible to develop a new sequence-based feature to predict DNA-binding sites in protein?

Likewise this leads to more detailed subquestions resolved in this thesis, which are:

- **RQ 2.1:** How could we extract this new feature through the means of information theory?
- **RQ 2.2:** How could we minimize the negative effect of a class imbalance between the number of binding and non-binding sites?

1.2. Scope of the Thesis

In this thesis, I firstly present a graph-based method to calculate rigid domains in proteins. This new method infers a binary labeling encoding whether a pair of amino acids belong to the same or to different domains. The algorithm consists of two stages. First of all, I create a protein graph constructed from spatial proximity of multiple conformational states. Due to the computational capacity, I reduce this graph to obtain its coarse-grained version via Louvain clustering algorithm [39]. Secondly, I label edges of the reduced graph through a line graph transformation along with generalized Viterbi algorithm [10]. For the evaluation, I illustrate how the algorithm proceeds through a segmentation of Adenylate Kinase. Moreover, I also benchmark this algorithm on 487 protein structures in the DynDom database, which results in a high agreement to the reference segmentations. Last but not least, I dis-

cuss detailed analyses of several proteins ranging from different scale transformations and compare them to other methods.

Regarding to the second project, I present a new feature extraction method to predict DNA-binding sites in proteins. Despite the rich literature on sequence-based feature extraction, it is useful to develop a new feature which assists the existing ones. In the feature engineering phrase, I extract a new feature based on the assumption that the amino acid distributions of binding and non-binding sites in a protein are essentially different. From the training data set, I compute a null background distribution of non-binding sites. Later, my new feature for each amino acid is a single value ranging from zero to one, calculated by a percentile transformation of a set of scores whose values are the weighted sum of Jensen–Shannon divergence (JSD) between amino acids distribution of sites as well as their neighbor sites and the null background distribution. Afterward, I incorporate the new sequence-based feature into existing ones and evaluate these via Random Forest (RF) classifier with five-fold cross validation.

1.3. Impact

During the course of this work, the results have been published in the following peer reviewed journal articles:

- L. Dang, T. Nguyen, M. Habeck, M. Gültas and S. Waack, "A graph-based algorithm for detecting rigid domains in protein structures," *BMC Bioinformatics* 22, 66 (2021). `https://doi.org/10.1186/s12859-021-03966-3`.
- L. Dang, C. Meckbach, and R. Tacke, S. Waack and M. Gültas "A Novel Sequence-Based Feature for the Identification of DNA-Binding Sites in Proteins Using Jensen–Shannon Divergence," *Entropy*, vol. 18, pp. 379, 2016. [Online]. Available: `https://www.mdpi.com/1099-4300/18/10/379`

Additionally, I also contribute to the following publication related to protein research topic:

- Z. Dong, K. Wang, L. Dang, M. Gültas, M. Welter, T. Wierschin, M. Stanke and S. Waack, "CRF-based models of protein surfaces improve protein-protein interaction site predictions," in *BMC Bioinformatics*, vol. 15, pp. 277, 2014.

Conferences and Seminars

The author have presented research results in the following seminar and conferences:

- Bioinformatics Seminar (Göttingen, December 2019): speaker
- German Conference on Bioinformatics (GCB 2019, Heidelberg): poster presentation
- Structural Biology Conference (2018, Murnau): poster presentation

- German Conference on Bioinformatics (GCB 2016, Berlin): poster presentation

1.4. Structure of the Thesis

Following the introduction, I reiterate biological and mathematical backgrounds in chapter 2 and 3. Next, I present methods utilized to find the rigidity in proteins as well as the new sequence-based features for DNA-binding sites prediction in proteins. Then, I evaluate my new methods through multiple experiments. Finally, I summarize and discuss new findings at the end of this thesis. For more information, the detailed contents of chapters are as follows:

- **Chapter 2** describes the necessary biological backgrounds of this thesis. In this, I present fundamental concepts such as proteins, protein interactions and bioinformatics resources one retrieves.
- In **Chapter 3**, I describe mathematics-related concepts used in this study, such as information theory, random forest classifier, the generalized Viterbi algorithm, Louvain algorithms as well as the outlier detection.
- **Chapter 4** consists of two parts where I present methods to detect important sites in proteins. In the Section **4.1**, I describe novel methods to calculate rigid domains in proteins which is important to understand how proteins move in a large scale. Section **4.2** presents a new sequence-based feature derived from information theory which is used to predict the DNA-binding sites in proteins.
- In **Chapter 5**, I present the evaluations of my approaches to identify important sites in proteins. This chapter comprises two subsections. First, I validate the effectiveness of my graph-based segmentation algorithm by evaluating it through various protein structures and benchmarking the method with 487 entries in DynDom dataset. In the second part of the chapter, I present the results of how I improve the DNA-binding sites in proteins prediction with my new information theoretic feature.
- **Chapter 6** offers elaborative discussion of the two methods I mentioned above. I analyze the choice of parameters and their contribution to the overall algorithms.
- Finally, in **Chapter 7**, I summarize the thesis and present my outlook for future work.

2. Biological Backgrounds

This chapter contains the biological foundations of molecular processes which are necessary to fully grasp the motivation of this thesis. I begin by introducing terms related to proteins, how these are synthesized in the gene expression and the classification of protein domain motions. After that, I present several sequence-based features for the DNA-binding sites in proteins prediction. At the end of this chapter, I present bioinformatics tools to obtain the data.

2.1. Proteins

Proteins are macromolecules made up of hundreds or thousands of smaller subunits called amino acids connected to one another in long chains. In general, amino acids are organic compounds consisting of functional groups amine ($-NH_2$) and carboxyl ($-COOH$) along with a side chain (R group) specific for each amino acid (Figure 2.1). Sequence of amino acids are connected to each other via peptide bonds, as illustrated in Figure 2.2.

Even though there are hundreds of different amino acids founded in nature, only twenty appear in proteins. These kinds of amino acids could be classified according to the structure and general chemical characteristics of their R groups such as polarity, electrical charge, aromaticity, and acidity.

Proteins are synthesised in cells through transcription and translation phases [40]. During transcription, a gene - a section of DNA in genome encoding a protein is converted into a template molecule named *mRNA* (messenger RNA) which is single stranded by RNA

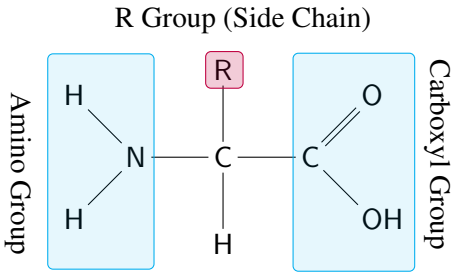

Figure 2.1.: Chemical structure of an amino acid with R group

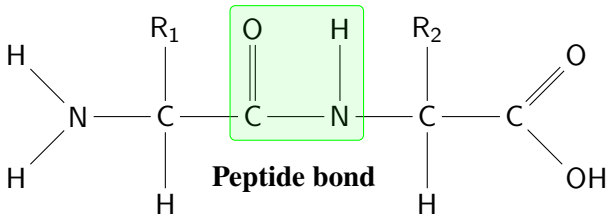

Figure 2.2.: Peptide bond between two amino acids

polymerases. Afterward, the translation phase or the reading of those *mRNAs* to produce proteins takes place in a ribosome which is a complex of proteins and *RNAs*. The order of amino acids added to a growing protein when it is synthesized depends on each *mRNA* comprised of a sequence of nucleotide A, U, C and G. In specific, a three-nucleotide sequence or a codon along the *mRNA* is translated to a specific amino acid through a condon amino acid mapping [41]. In addition, there are two extra start and stop codons to navigate the translation process.

2.2. Conformational Change in Proteins

Even though many static structures of proteins are determined, their biological functions are ultimately driven by their own motions [11]. To thoroughly study structural changes, it is rational to consider such conformational transitions as the combination of hinge and shear motions of rigid domains in proteins. Hinge motions, which are often occurred in large structural transformation, include the hinge regions functioning as the linker of several rotational protein domains. Those hinge regions consists of many residues which are involved in significantly structural changes, while other residues in rotating domains remain largely unchanged [22]. Shear motions, on the other hand, comprise of sliding movement of protein domains to each other. This sort of motion results in small conformational changes [22].

2.3. Features Extraction for DNA-binding Sites in Proteins Prediction

There are several features which could be extracted from sequences. In this thesis, I present three sequence-based features I utilized for the prediction of DNA-binding sites in proteins. Those features encode the evolutionary information, the secondary structure and characteristic properties of amino acids.

2.3.1. Position-specific Scoring Matrix

Position-specific scoring matrix (PSSM) is a method to encode the evolutionary information of amino acid sequences [36]. For each protein sequence, its corresponding multiple sequence alignment (MSA) is constructed by protein sequence searching tools such as HHblits [42] or PSI-BLAST [43]. A PSSM profile of a MSA associated with a protein length N is a $N \times 20$ matrix whose values are integer numbers. A positive score suggests a certain amino acid occurs more often by chance than the average. In contrast, a negative score indicates the corresponding amino acid unlikely occurs at that position.

The PSSM-based feature then encodes a statistic indicating the likelihood of certain amino acids occurred at a certain position. For each amino acid in the protein sequence, its PSSM-based feature is a 20D vector encoding the frequencies of twenty amino acids of this site.

2.3.2. Secondary Structure

Wu et al. [27] integrate the secondary structure (SS) information into the feature set for DNA-binding sites in proteins prediction. The secondary structure information could be extracted either from PDB files or predicted from a secondary structure prediction program such as PREDATOR [44]. For each amino acid in a protein sequence, its SS is classified into three categories involving an *alpha-helix*, a *beta-strand* and others which could be encoded through 3D vectors $(0,0,1)$, $(0,1,0)$ and $(1,0,0)$ respectively.

2.3.3. Orthogonal Binary Vector

The interfaces in DNA-protein complexes are mainly dominated by hydrogen bonds whose affinities strongly depend on the dipoles including such bonds and the structural complementation between amino acids and DNA helix grooves [45]. For the protein-protein interactions, Shen et al. [46] grouped twenty kinds of amino acids into seven groups according to their dipoles and volumes of the side chains using density-functional theory and molecular modeling approach. Wu et al. [27] later reduced this classification to six groups which was more suitable for the DNA-binding sites in proteins prediction. These classes are:

- Class 1: Ala, Gly, Val;
- Class 2: Ile, Leu, Phe, Pro;
- Class 3: Tyr, Met, Thr, Ser, Cys;
- Class 4: His, Asn, Gln, Tpr;
- Class 5: Arg, Lys;
- Class 6: Asp, Glu.

These amino acid groups are encoded as 6D orthogonal binary vectors (OBV). Each vector associated to each class comprises of six binary values such as $(0,0,0,0,0,1)$, $(0,0,0,0,1,0)$, $\cdots$, $(1,0,0,0,0,0)$ respectively.

It is important to note that the encoding method for either SS and OBV results in binary vectors which are orthogonal to each other. This orthogonality would prevent negative affects of non-orthogonal features [47] which are the distortion of the metric structure in original data space [48] and the redundant information [49].

2.4. Biological Information Sources

2.4.1. HH-suite

I ran the stand-alone *HHblits*, a program in *HH-suite* [50] (version v2) to create MSAs as well as PSSMs of proteins for the training and evaluating phases. The protein database I used to create MSAs is *uniprot20_2016_02* which is available at http://wwwuser.gwdg.de/ compbiol/data/hhsuite/databases/hhsuite_dbs/old-releases.

2.4.2. Protein Data Bank

The Protein Data Bank (PDB) is a leading database for the structural biology research containing all 3D structures as well as functional notations of proteins, nucleic acids, complexes among proteins and nucleic acids, and other biological molecules which are determined through experiments. The PDB database is organized and maintained at the Research Collaboratory for Structural Bioinformatics (RCSB) and freely available at rcsb.org. In this database, proteins are indexed via PDB entries which are 4D strings, in each comprising of either Roman alphabet (A to Z) or Arabic digits (0 to 9) such as "1ake" (Adenylate kinase).

All structural information regarding to a protein and its complexes could be retrieved at the PDB database. One could find a comprehensive format of PDB file via *Protein Data Bank Contents Guide* [51]. To summarize, a standard PDB file of a protein entry contains (i) name of authors, (ii) literature references, (iii) details of experiments to determine protein structures, (iv) atomic coordinates of the complex, (v) primary and secondary structure such as α-helix and β-sheet, (vi) binding and active sites, inter alia. One could either manually parse the PDB files or use publicly available packages such as *BioJava* [52] or *BioPython* [53] to access this information.

2.4.3. Non-redundant DynDom Database

The non-redundant database of protein domain movements [54], available at dyndom.cmp.uea.ac.uk/dyndom/, is a collection of protein motions grouped into families

to remove the redundancy. From this database web-service, users could search and browse protein families where they could access notations of protein domains in different conformational states.

2.4.4. TM-Score

The template modeling score (TM-score), developed by *Zhang* and *Skolnick* [55], is a means to assess the similarity of two protein structures. In [56], Zhang et al. state that this score is superior to the traditional metrics such as root-mean-square deviation (RMSD) [16] in two folds : (i) it achieves more sensitivity to the global fold similarity than to the local structural variations by readjusting weights of small and large distance errors, (ii) it normalizes the distance errors and enables to work with any structure pair by introducing a length-dependent scale.

Users could run *TM-score* via standalone software or web-service at zhanglab.ccmb.med.umich.edu/TM-score.

3. Mathematical Backgrounds

This chapter recapitulates some important mathematical and informatics concepts used in this thesis. First, I present the information theory and information theoretic-related divergences. Further, I describe some of the machine learning methods such as RF and CRF which I applied to my research projects. Finally, I deliver all backgrounds related to the algorithm developed to calculate rigid domains in proteins.

3.1. Information Theory

Information theory was firstly proposed by Shannon to investigate the limitation on signal processing and how much redundancy can be removed to achieve the optimal compression in communication operations [57]. This field is interdisciplinary among mathematics, statistics, computer sciences and electrical engineering and has been successfully applied to bioinformatics in various areas such as DNA-binding sites motif [58], structurally important sites in proteins [59], prediction of protein functional residues [60, 61], and symbolic sequence analysis [62].

In this section, I recapitulate the fundamental concepts in information theory such as entropy, conditional entropy and mutual information. More details can be found in [63]

3.1.1. Shannon Entropy

Given a discrete random variable $\mathbf{X}$ and a set of its possible outcomes $\mathbf{D_X}$, a *Shannon Entropy* is notated as $H(\mathbf{X})$ and defined as

$$H(\mathbf{X}) = -\sum_{x \in \mathbf{D_X}} p(x) \cdot \log p(x) \tag{3.1.1}$$

This Shannon entropy quantifies the mean of $\log p(x)$; or in another words it is the smallest number of bits needed to eliminate the uncertainty of the associated distribution. This Shannon entropy is generalized to continuous random variables by replacing the *sum* notation with the integral over the set of value of its random variable.

3.1.2. Conditional Entropy

The conditional entropy of a random variable X given another random variable Y is defined as

$$H(\mathbf{X}|\mathbf{Y}) = -\sum_{x\in\mathbf{D_X}}\sum_{y\in\mathbf{D_Y}} p(x,y)\cdot\log p(x|y) \tag{3.1.2}$$

The conditional entropy tells us how much entropy of a certain random variable is reduced if we have known the other information from the other random variable. Thus, $H(\mathbf{X}|\mathbf{Y})$ equals zero if $\mathbf{Y}$ implies $\mathbf{X}$ and equals to $H(\mathbf{X})$ if $\mathbf{X}$ and $\mathbf{Y}$ are independent.

3.1.3. Mutual Information

Mutual information between random variables $\mathbf{X}$ and $\mathbf{Y}$ measures the uncertainty which could be reduced given the present of another source of information. Formally, it is defined as

$$\begin{aligned} I(\mathbf{X};\mathbf{Y}) &= H(\mathbf{X}) - H(\mathbf{X}|\mathbf{Y}) \\ &= H(\mathbf{Y}) - H(\mathbf{Y}|\mathbf{X}) \end{aligned} \tag{3.1.3}$$

Mutual information between two random variables would be zero if they are independent.

3.1.4. Kullback–Leibler Divergence

The Kullback–Leibler (KL) divergence is a directed divergence between two probability distributions. This divergence is asymmetric and could be interpreted as the amount of information one loses when he or she uses one probability distribution to approximate the other. Formally, given two probability distributions $\mathbf{P}$ and $\mathbf{Q}$ defined on the same probability space X, the KL divergence from $\mathbf{Q}$ to $\mathbf{P}$ is defined as:

$$D_{KL}(\mathbf{P}||\mathbf{Q}) = -\sum_{x\in\mathrm{X}} p(x)\cdot\log\frac{q(x)}{p(x)} \tag{3.1.4}$$

3.1.5. Jensen–Shannon Divergence

The JSD, derived from KL divergence with substantial changes, measures the similarity between two probability distributions. Unlike KL divergence, JSD is symmetric and its square root is a metric. Similar to KL divergence, the JSD between two probability distributions over probability space X is defined as:

$$JSD(\mathbf{P}||\mathbf{Q}) = \frac{1}{2}\cdot(D_{KL}(\mathbf{P}||\mathbf{M}) + D_{KL}(\mathbf{Q}||\mathbf{M})) \tag{3.1.5}$$

where $\mathbf{M} = \frac{\mathbf{P}+\mathbf{Q}}{2}$. The result of Equation 3.1.5 is a non-negative real number bounded by $+1$, occurred when the two distributions are completely different, e.g. $\mathbf{P}$ and $\mathbf{Q}$ are different constant random variables.

3.2. Machine Learning Techniques

Machine learning is a study of statistical models which can learn a specific task based on patterns and inferences. Machine learning techniques mainly consist of supervised and unsupervised approaches. In the supervised statistical learning, the parameters of the models are adjusted according to the sample labels in the training set. On the other hand, the training set in unsupervised learning has no label. In the scope of this study, I present two methods in supervised learning methods which are Random Forest (RF) and Conditional Random Field (CRF).

3.2.1. Random Forest

Random Forest is a statistical learning method invented by Breiman which consists of many substantial modified classification trees [64]. In this subsection, I firstly review the concept of tree-based learning methods such as Classification and Regression Tree. Finally, I present the Random Forest machine learning algorithm.

Classification and Regression Tree

Classification and Regression Tree (CART) is a tree-based learning method where a decision tree is built up according to its training data set. In bioinformatics research, CART has been successfully applied to detect emerging patterns for cancer diagnosis [65] and to improve the analysis of high-throughput genomic data [66]. For the overview of the application of CART in Bioinformatics, readers may have a look at [67, 68]. The notations and description of the CART algorithm in this thesis are taken from [69].

Algorithm 3.1 Pseudo code of CART algorithm.

Convention: A pair (p,v): $p \in \mathbb{N}$ is an index of a feature dimension and $v \in \mathbb{R}$ is a value of a certain sample at this dimension.
Input: Sample set S = $\{\mathbf{X},\mathbf{y}\}$, where $\mathbf{X} \in \mathbb{R}^{M\times P}$ and $\mathbf{y} \in \{1,2,..J\}^M$, M is a number of training samples, P is the number of features and J is the number of labels.
// Each row of a matrix $\mathbf{X}$ is a training sample along with its binary label in $\mathbf{y}$.
while not stopping criteria **do**
    // The stopping criteria will be explained later.
    Choose a pair (p,v) such that we achieve the "greatest separate"
    // The term "greatest separate" will be explained later.
    Split the node according to the threshold value v at the p^{th} feature dimension.
    Apply the procedure to the new left and right nodes.
end while
Output: A classified tree

The *stopping criteria* indicates the algorithm to stop either if the tree is homogeneous (all data points have an identical label) or the number of data points is below a certain threshold.

The key idea of splitting is to obtain the *greatest separation*, or in another words, to create child nodes which are the *purest*. Let us denote $i(t)$ be the impurity function of a node t in the tree. The main target is to search the split point where the decrease of the impurity $\Delta i(t) = i(t) - \mathbb{E}[i(t_{childs})]$ is maximal (t_{childs} consists of the right and left node of t). That means

$$p^*, v^* = \underset{p=1..P,\ v\in\mathbb{N}}{\arg\max} \big(\Delta i(t)\big) \tag{3.2.1}$$

where p is the index of a feature dimension and v is the splitting value on that dimension.

For the discrete value of the label $\mathbf{y}$, the impurity of a node could be calculated through many methods such as entropy, Gini and Twoing [70]. I will present the Gini impurity as an example of how to calculate the impurity.

The Gini impurity, used by the CART algorithm, quantifies the likelihood of incorrectly labelling a random element in a set based on the class distribution of that set. Mathematically, the Gini impurity is computed by $GI(p) = \sum_{i=1}^{J} p(i)\sum_{j\neq i} p(j) = \sum_{i=1}^{J} p(i)(1-p(i))$ where J is the number of classes in a dataset.

Random Forests Algorithm

The description and notations of RF classifiers are taken from the original source [64].

Algorithm 3.2 Pseudo code of Random Forest classifiers.

1: *Input*: $S = \left\{X^{(i)}, y^{(i)} : i = 1..M\right\}$ is a training set containing M samples. $X^{(i)} \in \mathbb{R}^P$ is an i-th training instance with P dimensional feature. $y^{(i)}$ is a label of $X^{(i)}$, typically belong to a binary set $\{0,1\}$.

2: **for** b = 1 **to** B **do**

3: // B: number of tree in the forest.

4: Randomly draw with replacement N bootstrap samples from S (typically $N \sim 2/3M$).

5: Recursively construct a tree $T^{(b)}$ from those bootstrap N samples by the following procedure:

- Randomly selecting m out of P feature variables,
- Searching the best splitting point (as mentioned in CART algorithm) among those m features,
- Splitting the node into left and right nodes.

6: **end for**

7: *Output*: Return $\left\{T^{(b)}\right\}_{b=1}^{B}$.

Given new instances for the class prediction, their predicted labels are the majority vote of B trees which are the outcome of the RF classifier.

3.2.2. Conditional Random Field

Suppose we have a site graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N vertices $\mathcal{V} = \{1,2,..,N\}$ labeled by an element from a finite set $\mathcal{B}$ (typically $\mathcal{B} = \{0,1\}$). Let $\mathcal{O}$ be a finite set including values from the observation. $\mathcal{E}$ is a set of edges in $\mathcal{G}$. The neighborhood set of a vertex $i \in \mathcal{V}$ denoted as $\mathcal{N}_i$ consists of vertices which link to vertex i. For any subset $I \subseteq \mathcal{V}$, y_I is a label sequence of the set I while y without subscript is the sequences of labels for a whole graph. In addition, for any $e \in \mathcal{E}$, y_e is a pair of labels of two vertices of e.

A pair $(\mathbf{X}, \mathbf{Y})$ consists of observations $\mathbf{X} \in \mathcal{O}^N$ and sequence labels $\mathbf{Y} \in \mathcal{B}^N$ realizes an exponential model if the conditional probability $p(y|x)$ of all pairs (x,y) is

$$p(y|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \exp\left(\sum_{s=1}^{c} \sum_{|I|=s} \left(\Psi^{(s)}(y_I, \mathbf{X})\right)\right) \tag{3.2.2}$$

where $Z(\mathbf{X})$ is the normalization factor and $\sum_{|I|=s}$ is a sum of cliques I with size of s in the graph $\mathcal{G}$ and c represents the number of nodes of the largest clique. $\Psi^{(s)}(y_I, \mathbf{X})$ denotes the feature function of a graph defined on the clique size s. This feature-based exponential model indeed coincides with the class of CRF where every vertex i is conditionally

independent to other vertices outside N_i given the neighborhood set N_i.

A pairwise CRF is a simplification version of CRF where one only considers clique size of one (vertices) and two (edges), i.e. $s = 2$. Thus, the conditional probability in Equation (3.2.2) becomes:

$$p(y|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \exp\left(\sum_{i\in\mathcal{V}} \Psi^{(1)}(y_i, \mathbf{X}) + \sum_{(i_1,i_2)\in\mathcal{E}} \Psi^{(2)}(y_{i_1}, y_{i_2}, \mathbf{X})\right) \tag{3.2.3}$$

More precisely, the feature functions defined on vertices and edges could be decomposed as the linear combination of other functions. Thus, the Equation (3.2.3) could be rewritten as:

$$p(y|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \exp\left(\sum_{i\in\mathcal{V}}\sum_{k=1}^{K} \alpha_k f_k(y_i, \mathbf{X}) + \sum_{(i,j)\in\mathcal{E}}\sum_{l=1}^{L} \beta_l g_l(y_i, y_j, \mathbf{X})\right) \tag{3.2.4}$$

$$= \frac{1}{Z(\mathbf{X})} \exp\left(\sum_{i\in\mathcal{V}} \alpha^T f(y_i, \mathbf{X}) + \sum_{(i,j)\in\mathcal{E}} \beta^T g(y_i, y_j, \mathbf{X})\right) \tag{3.2.5}$$

where f and g are the vector of feature functions applied on vertices and edges in the graph respectively. α and β are real-valued parameter vectors and $\alpha^T \& \beta^T$ are their transpose vectors.

Inference

Suppose that the vector of parameters (α, β) is given, the estimation of labels y for the whole graph is obtained by solving a following optimization problem

$$y^* = \arg\max_{y\in\mathcal{B}^N} p(y|\mathbf{X}) \tag{3.2.6}$$

$$= \arg\max_{y\in\mathcal{B}^N} \left(\sum_{i\in\mathcal{V}} \alpha^T f(y_i, \mathbf{X}) + \sum_{(i,j)\in\mathcal{E}} \beta^T g(y_i, y_j, \mathbf{X})\right) \tag{3.2.7}$$

where the term $Z(\mathbf{X})$ can be ignored because it is not a function of y.

If $\mathcal{G}$ is a tree-like graph, this inference problem could be solved exactly through a Viterbi algorithm for tree-like structures [71] or an argmax-version of Felsenstein's tree-pruning algorithm [72].

In general, solving the Equation (3.2.7) for an arbitrary graph is infeasible. However, there are a few approximation methods to resolve such problem such as Loopy Belief Propagation [73] and the generalized Viterbi algorithm [10] which I discuss in details in Section 3.3.

Learning

In the learning phrase, we are given a sample learning set $\{ (X_d, y_d) \}_{d=1}^{N}$, our goal is to find model parameters (α^*, β^*) such as

$$\alpha^*, \beta^* = \arg\max_{\alpha,\beta} \prod_{d=1}^{N} p(y_d | X_d, \alpha, \beta) \tag{3.2.8}$$

$$= \arg\max_{\alpha,\beta} \sum_{d=1}^{N} \left(\sum_{i \in \mathcal{V}} \alpha^T f(y_{d,i}, X_d) + \sum_{(i,j) \in \mathcal{E}} \beta^T g(y_{d,i}, y_{d,j}, X_d) \right) \tag{3.2.9}$$

This problem could be solved by taking the derivatives of the likelihood with respective to α and β. Unfortunately, these computations are intractable because they require an exponentially large number of summations. To overcome this issue, one could utilize the pseudo maximum likelihood approximation where one only takes the Markov blanket of a vertex into account.This results in a convex problem and thus the maximal points could be found via numerical methods such as gradient decent or online large margin techniques [74].

3.3. The Generalized Viterbi Algorithm

Let us denote $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a neighborhood graph with a binary labeled vertex set $\mathcal{V} = \{v_i : i = 1, 2, \ldots, N\}$ and an edge set $\mathcal{E}$. For any assignment/label $y \in \{0,1\}^N$ and a subset $\mathcal{V}' \subseteq \mathcal{V}$, we denote $y_{\mathcal{V}'}$ be the sub-assignment of y on the subset $\mathcal{V}'$. According to the pairwise CRF, the logarithm of quality function (*logquality*) defined on $\mathcal{G}$ is defined as

$$F(y_{\mathcal{V}} | \mathcal{V}, \mathcal{E}) = \sum_{v \in \mathcal{V}} \Psi^{(1)}(v, y_v) + \sum_{(v_1, v_2) \in \mathcal{E}} \Psi^{(2)}\left(v_1, v_2, y_{v_1, v_2}\right) \tag{3.3.1}$$

where $\Psi^{(1)}$ and $\Psi^{(2)}$ are feature functions defined on vertices and edges respectively.

The generalized Viterbi algorithm is an heuristic approach to determine the most probable label $y^*_{\mathcal{G}}$ given a well defined quality function

$$y^*_{\mathcal{G}} = \arg\max_{y_{\mathcal{G}} \in \{0,1\}^N} F(y_{\mathcal{V}} | \mathcal{V}, \mathcal{E}) \tag{3.3.2}$$

The condition of F on set of vertices and edges could be skipped when it is clear which vertices and edges are involved. Thus, the Equation (3.3.2) becomes $y^*_{\mathcal{G}} = \arg\max_{y_{\mathcal{G}} \in \{0,1\}^N} F(y_{\mathcal{V}})$. The full description of the generalized Viterbi algorithm is described in [10]. In this section I present a pseudo code as well as a small example and explain how this algorithm works step by step.

3.3.1. Pseudo Algorithm

I denote $\mathcal{H}, \mathcal{B}_\mathcal{H}, \mathcal{I}_\mathcal{H}, \mathcal{N}_\mathcal{B}$ be the *history set*, *boundary set* of the history set, *interior set* of the history set, and *neighborhood set* of the boundary set respectively. The history set contains all vertices that the algorithm has processed. The boundary set is a subset of the history set containing vertices having neighbors not contain in the history set. The interior set is the complement of the boundary set in the history set, i.e. $\mathcal{I}_\mathcal{H} = \mathcal{H} \setminus \mathcal{B}_\mathcal{H}$. The neighborhood set consists of vertices which are neighbors of the boundary set but do not belong to the history set. To avoid the notation cumbersome, I neglect the subscript of the history set when it is clear in the context. Additionally, I use a table $\mathbf{T}$ to store all maximal values of the $logquality$ function defined on the history set according to the interior set. Follows are a pseudo code and an example of the generalized Viterbi algorithm.

Pseudo Code

Algorithm 3.3 Pseudo code of the generalized Viterbi algorithm.

1: $\mathcal{H} \leftarrow \{\}$ // the history set is initialized as an empty set.

2: Thus, $\mathcal{B}_\mathcal{H} \leftarrow \{\}$, $\mathcal{I}_\mathcal{H} \leftarrow \{\}$ and $\mathcal{N}_\mathcal{B} \leftarrow \mathcal{V}$.

3: **while** $\mathcal{H} \neq \mathcal{V}$ **do**

4:

$$v^* = \arg\min_{v \in \mathcal{N}_\mathcal{B}} \|\mathcal{N}_{\mathcal{H} \cup v}\| \tag{3.3.3}$$

where $\|\cdot\|$ is the cardinality of a set

5: $\mathcal{H} \leftarrow \mathcal{H} \cup v^*$

6: $\mathcal{B}_\mathcal{H}, \mathcal{I}_\mathcal{H} \leftarrow$ updated according to $\mathcal{H}$

7: $\mathcal{N}_\mathcal{B} \leftarrow$ updated according to $\mathcal{B}_\mathcal{H}$

8: For every labels of the boundary set $\ell \in 2^{\mathcal{B}_\mathcal{H}}$, we calculate:

$$\mathbf{T}(y_{\mathcal{B}_\mathcal{H}} = \ell) = \max_{y_{\mathcal{I}_\mathcal{H}}} F(y_\mathcal{H} | y_{\mathcal{B}_\mathcal{H}} = \ell)$$

9: **end while**

Using dynamic programming, the required computational units to calculate the table $\mathbf{T}$ at t-th iteration requires only twice the size of $\mathbf{T}$ in previous iteration. Yet one would need $2^{|\mathcal{B}|}$ entries to store the results of all possible labels. Thus, the algorithm becomes infeasible when the boundary set is large. To prevent such exponential growth, the authors just keep a certain number of entries, for example 10000, whose values are maximal [10].

An Example

Suppose we are given a graph with six vertices and nine edges, as shown in the Figure 3.1.

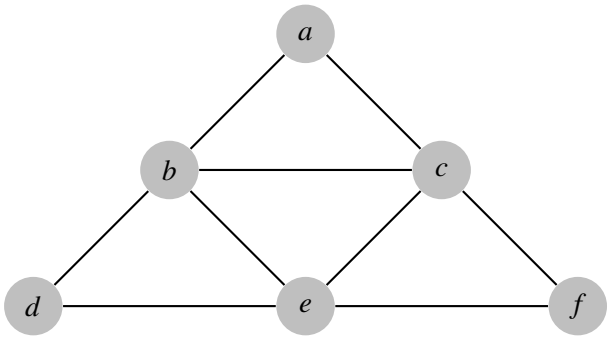

Figure 3.1.: An undirected graph.

The values of the feature functions defined on vertices ($\Psi^{(1)}$) and on edges ($\Psi^{(2)}$) are described in the Table 3.1 and 3.2 respectively.

Vertex \ Label	0	1
a	-1	1
b	0	0
c	1	-1
d	-1	1
e	-1	1
f	1	-1

Table 3.1.: Values of feature function $\Psi^{(1)}$ defined on vertices.

Edge \ Label	00	01	10	11
a-b	-1	1	-1	0
a-c	0	1	-1	-1
b-c	-1	1	-1	1
b-d	-1	-1	0	0
b-e	0	1	0	1
c-e	-1	-1	-1	1
c-f	1	0	1	0
d-e	0	-1	0	1
e-f	0	-1	1	0

Table 3.2.: Values of feature function $\Psi^{(2)}$ defined on edges.

At the initial step, the *history set* $\mathcal{H}$, *boundary set* $\mathcal{B}$ and *interior set* $\mathcal{I}$ are set to be empty.

$\mathcal{H} \leftarrow \{\}$, $\mathcal{B} \leftarrow \{\}$. Thus the *neighborhood set* $\mathcal{N} \leftarrow \{a,b,c,d,e,f\}$.

Firstly, the algorithm chooses a vertex with a minimal neighborhood, saying vertex a. Thus, the history and boundary set are updated as: $\mathcal{H} \leftarrow \{a\}$, $\mathcal{B} \leftarrow \{a\}$. Consequently, the neighborhood set of the boundary set becomes $\mathcal{N} \leftarrow \{b,c\}$

The values of the *logquality* function F defined on the boundary set are:

$y_{\{a\}}$	0	1
$F(\{a\})$	-1	+1

Table 3.3.: Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{a\}$.

Afterward, we choose a vertex in set $\mathcal{N}$ to add to the history set, saying the vertex b. Thus, we have $\mathcal{H} \leftarrow \{a,b\}$, $\mathcal{B} \leftarrow \{a,b\}$, and $\mathcal{N} \leftarrow \{c,d,e\}$. The table T becomes:

$y_{\{a,b\}}$	00	01	10	11
$F(\{a,b\})$	-2	0	0	+1

Table 3.4.: Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{a,b\}$.

In the next step, the algorithm chooses a vertex c due to the minimal boundary set requirement. Thus, we have $\mathcal{H} \leftarrow \{a,b,c\}$, $\mathcal{B} \leftarrow \{b,c\}$, and $\mathcal{N} \leftarrow \{d,e,f\}$. The table T

becomes:

$y_{\{b,c\}}$	00	01	10	11
$\max_a F(\{a,b,c\})$	$-1_{(a=1)}$	$-1_{(a=\cdot)}$	$0_{(a=\cdot)}$	$+1_{(a=0)}$

Table 3.5.: Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{b,c\}$.

where "$a = \cdot$" means vertex a could be assigned to any label.

Then, for instance, we choose the vertex d. Thus we have $\mathcal{H} \leftarrow \{a,b,c,d\}$, $\mathcal{B} \leftarrow \{b,c,d\}$ and $\mathcal{N} \leftarrow \{e,f\}$. The table T becomes:

$y_{\{b,c,d\}}$	000	001	010	011	100	101	110	111
$\max_a F(\{a,b,c,d\})$	$-3_{(a=1)}$	$-1_{(a=1)}$	$-3_{(a=\cdot)}$	$-1_{(a=\cdot)}$	$-1_{(a=\cdot)}$	$+1_{(a=\cdot)}$	$0_{(a=0)}$	$+2_{(a=0)}$

Table 3.6.: Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{b,c,d\}$.

It is important to note that we do not need to go through all combination labels of $\{a,b,c,d\}$ to calculate $\max_a F(\{a,b,c,d\})$, but instead

$$\max_a F(\{a,b,c,d\}) = \max_a F(a,b,c) + \Psi^{(1)}(d) + \Psi^{(2)}(b,d)$$

where we could reuse the value of $\max_a F(a,b,c)$ in the previous iteration.

In the next iteration, the algorithm adds the vertex e into the history set $\mathcal{H}$ according to the condition in Equation (3.3.3). Thus, we have $\mathcal{H} \leftarrow \{a,b,c,d,e\}$, $\mathcal{B} \leftarrow \{c,e\}$, and $\mathcal{N} \leftarrow \{f\}$. The table T becomes:

$y_{\{c,e\}}$	00	01	10	11
$\max_{a,b,d} F(\{a,b,c,d,e\})$	$-2_{(a=0,b=1,d=1)}$	$+2_{(a=0,b=1,d=1)}$	$0_{(a=0,b=1,d=1)}$	$6_{(a=0,b=1,d=1)}$

Table 3.7.: Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{c,e\}$.

Similarly, we calculate values of table **T** as:

$$\max_{a,b,d} F(\{a,b,c,d,e\}) = \max_a F(\{a,b,c,d\}) + \Psi^{(1)}(e) + \Psi^{(2)}(b,e) + \Psi^{(2)}(c,e) + \Psi^{(2)}(d,e)$$

where we could reuse the value of $\max_a F(\{a,b,c,d\})$ in the previous iteration.

Finally, the algorithm adds the vertex f to $\mathcal{H}$. Consequently, we have $\mathcal{H} \leftarrow \{a,b,c,d,e,f\}$, $\mathcal{B} \leftarrow \{f\}$, and $\mathcal{N} \leftarrow \{\}$. The table T becomes

$y_{\{f\}}$	0	1
$\max_{a,b,c,d,e} F(\{a,b,c,d,e,f\})$	$9_{a=0,b=1,c=1,d=1,e=1}$	$5_{a=0,b=1,c=1,d=1,e=1}$

Table 3.8.: Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{f\}$.

Thus, the maximum value of $F(\mathcal{V})$ is 9 when $a = 0, b = 1, c = 1, d = 1, e = 1, f = 0$

3.4. Constant Potts Model in Louvain Algorithm

The Louvain algorithm is a method used to detect communities from the large network [75]. The description and insights of Louvain algorithm is out of the scope of this study, nonetheless one can be found at [39, 75]. In this section, I will present one of the very well-known frameworks used in Louvain algorithm and discuss its parameters. The following notations and models are taken from [39].

Notation

Let us denote a connected graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = N$ vertices and $|\mathcal{E}| = K$ edges. A corresponding adjacency matrix $\mathbf{A}$ of graph $\mathcal{G}$ is a square matrix where $\mathbf{A}_{ij} = 1$ if there is an edge connecting vertex i and j, or 0 otherwise. An element w_{ij} of a weight matrix $\mathbf{W}$ of the graph $\mathcal{G}$ encodes the (i,j) edge's weight. Finally, the community or label of a vertex i is denoted as σ_i. Similarly, $\sigma_{\mathcal{G}}$ denotes the label for every vertex in $\mathcal{G}$. In order to keep the formula short, the notation i could either stand for the vertex or its vertex's index.

The general assumption is that the connections within communities should be more frequent than those between communities. Thus, we would reward connections within communities and penalize missing links within communities [76]. Consequently, we define an identity according to that idea

$$\mathcal{H}(\sigma_{\mathcal{G}}) = -\sum_{ij,i<j} (a_{ij}\mathbf{A}_{ij} - b_{ij}(1-\mathbf{A}_{ij}))\mathbb{1}(\sigma_i = \sigma_j) \tag{3.4.1}$$

where $a_{ij}s$ and $b_{ij}s$ are non-negative parameters. Additionally, $\mathbb{1}(\sigma_i = \sigma_j)$ is an indicator function whose value is $+1$ if $\sigma_i = \sigma_j$ or 0 otherwise.

The desirable partition of G corresponds to the minimal value of $\mathcal{H}$. Mathematically, it

is defined as

$$\sigma_{\mathcal{G}}^{*} = \arg\min_{\sigma_{\mathcal{G}}} \mathcal{H}(\sigma_{\mathcal{G}}) \tag{3.4.2}$$

The choice of the parameters $a_{ij}s$ and $b_{ij}s$ depends on what type of communities we would wish to detect. In the scope of this thesis, I only discuss the choice of Constant Potts Model (CPM).

Constant Potts Model

In [39], they define $a_{ij} = w_{ij} - b_{ij}$ and $b_{ij} = \gamma$. Thus, the Equation 3.4.1 becomes:

$$\begin{aligned}
\mathcal{H}(\sigma_{\mathcal{G}}) &= -\sum_{ij,i<j} ((w_{ij} - b_{ij})\mathbf{A}_{ij} - \gamma(1 - \mathbf{A}_{ij}))\mathbb{1}(\sigma_i = \sigma_j) \\
&= -\sum_{ij,i<j} (w_{ij}\mathbf{A}_{ij} - \gamma)\mathbb{1}(\sigma_i = \sigma_j) \\
&= -\sum_{ij,i<j} (w_{ij}\mathbf{A}_{ij}\mathbb{1}(\sigma_i = \sigma_j) - \gamma\mathbb{1}(\sigma_i = \sigma_j))
\end{aligned} \tag{3.4.3}$$

Suppose that we are given a partition $\mathbf{C}$ of $\mathcal{G}$. Instead of summing over vertices i and j in the whole network, we could rewrite the equation 3.4.3 as the sum over its partition (note that $\mathbb{1}(c = c') = 0$ if $c \neq c'$,):

$$\begin{aligned}
\mathcal{H}(\sigma_G) &= -\sum_{c\in\mathbf{C}} \sum_{i<j\in c} (w_{ij}\mathbf{A}_{ij}\mathbb{1}(\sigma_i = \sigma_j) - \gamma\mathbb{1}(\sigma_i = \sigma_j)) \\
&= -\sum_{c\in\mathbf{C}} \left(\sum_{i<j\in c} w_{ij}\mathbf{A}_{ij}\mathbb{1}(\sigma_i = \sigma_j) - \sum_{i,j\in c} \gamma\mathbb{1}(\sigma_i = \sigma_j) \right) \\
&= -\sum_{c\in\mathbf{C}} \left(\sum_{i<j\in c} w_{ij}\mathbf{A}_{ij}\mathbb{1}(\sigma_i = c)\mathbb{1}(\sigma_j = c) - \sum_{i<j\in c} \gamma\mathbb{1}(\sigma_i = c)\mathbb{1}(\sigma_j = c) \right) \\
&= -\sum_{c\in\mathbf{C}} \left(e_c - \gamma n_c^2\right)
\end{aligned} \tag{3.4.4}$$

where $e_c = \sum_{i,j\in c} w_{ij}A_{ij}\mathbb{1}(\sigma_i = c)\mathbb{1}(\sigma_j = c)$ is the weighted sum of all edges in a community c and $n_c = \mathbb{1}(\sigma_i = c) = \mathbb{1}(\sigma_j = c)$ is the number of vertices in c. By minimizing $\mathcal{H}$, the algorithm tries to search for a community with dense internal connections while at the meantime relative small size [39]. The parameter γ balances these two key factors.

Suppose that $\{r,s\}$ is a partition of c and $e_{r\leftrightarrow s}$ is the weighted sum of edges connecting

r and s. From Equation 3.4.4, we have the value for a single community c as

$$\begin{aligned}\mathcal{H}(\sigma_c) &= -(e_c - \gamma n_c^2) \\ &= -((e_r + e_s + e_{r\leftrightarrow s}) - \gamma(n_r + n_s)^2) \\ &= -(e_r - \gamma n_r^2 + e_s - \gamma n_s^2 + e_{r\leftrightarrow s} - 2\gamma n_r n_s) \\ &= \mathcal{H}(\sigma_r) + \mathcal{H}(\sigma_s) + 2\gamma n_r n_s - e_{r\leftrightarrow s}\end{aligned} \tag{3.4.5}$$

It is important to note that if $2\gamma n_r n_s - e_{r\leftrightarrow s} > 0$ or $\gamma > \frac{e_{r\leftrightarrow s}}{2\gamma n_r n_s}$, we have $\mathcal{H}(\sigma_c) > \mathcal{H}(\sigma_r) + \mathcal{H}(\sigma_s)$. Thus, the community c should be split into r and s. The quantity $\frac{e_{r\leftrightarrow s}}{2\gamma n_r n_s}$ is indeed the density links between communities r and s. From that point of view, we could consider the constant γ as a threshold to split a big community into two smaller ones. In short, if $\gamma = \min_{i,j} A_{ij} w_{ij}$, the whole network will be never split, and thus the optimal solution is one big network. On the other hand, if $\gamma = \max_{i,j} A_{ij} w_{ij}$, the algorithm tends to split every community until all communities have only one vertex. Thus, it is reasonable to choose γ between $\min_{i,j} A_{ij} w_{ij}$ and $\max_{i,j} A_{ij} w_{ij}$.

3.5. Outliers Detection

Given a sequence of N real numbers $\mathcal{D} = [x_1, x_2, .., x_N]$, the robust variant of *Z score* [77] of each element is defined as:

$$\widetilde{Z}_i = 0.6745 \frac{x_i - MED}{MAD} \tag{3.5.1}$$

where MED is the sample median and MAD the sample median absolute deviation of $\mathcal{D}$ whose formula is $MAD = median(|x - MED|), x \in \mathcal{D}$. The constant 0.6745 is the $3/4$ quantile of the standard normal distribution arisen for consistency.

Iglewicz and Hoaglin [77] suggested ± 3.5 as a cut-off value, yet this is indeed the matter of choice. In this study, I use this default suggested threshold. Hence, I say an element x_i is an outlier according to $\mathcal{D}$ if and only if its absolute value of *Z score* defined in Equation 3.5.1 is greater than 3.5.

4. Probabilistic Models to Detect Important Sites in Proteins

In this chapter, I present my methods to identify important sites in proteins in different levels. In Section 4.1, I present my graph-based method to calculate rigid domains in proteins which is helpful for the study of protein transformations in a large-scale. Afterward, I describe the sequence-based feature extraction which is able to improve the prediction of DNA-binding sites in proteins (Section 4.2).

4.1. Identification of Rigid Domains in Proteins using a Graph-based Model

As mentioned in the Introduction section, many methods identifying rigid domains in proteins suffer from tuning parameters such as the number of rigid domains in proteins. Here I propose a new method which does not require the number of rigid domains as the parameter. The key idea is that I devise an algorithm binarily determining if a pair of amino acids belong to identical or to different rigid domains. The content of this section is organized as the follows. First, I present protein graph construction methods which encode the variability of multiple conformations of a protein. Second, I reduce the complexity of a protein graph by grouping vertices which are densely connected. Then, I describe a line graph transformation method used for the prediction of edges' labels. Afterward, I explain how to use the outliers detection mentioned in Section 3.5 to define the feature functions for vertices and edges in the framework of conditional random fields. Final, I provide an pseudo code for the method as well as the post-processing step which is necessary to avoid the advents of very small fragments.

The terminologies, contents, and structure of this section follow our published paper [13] (see Appendix A.1).

4.1.1. Protein Graph Constructions

Given a protein length N with M conformations, I encode each conformational state of the protein by a $N \times 3$ matrix $X \in \mathbb{R}^{N\times 3}$. Each row in matrix X is a $3D$ coordinate of a representative atom (typically $C\alpha$ *atom*). By convention, $X_n^{(m)}$ is a position of the n-th atom

in the m-th conformation. For each conformational state $X^{(m)}$, I calculate a symmetric $N \times N$ distance matrix $D^{(m)}$:

$$D_{i,j}^{(m)} = \|X_i^{(m)} - X_j^{(m)}\| \qquad (i,j = 1,2,\ldots,N), \tag{4.1.1}$$

where $\|\cdot\|$ is the Euclidean norm.

I encode a protein fluctuation across M conformational states through a protein graph $\mathcal{PG} = (\mathcal{PV}, \mathcal{PE})$ whose vertices $\mathcal{PV}$ are representative atoms of amino acids and edges $\mathcal{PE}$ contain the relationship information among vertices. There are several reasonable options to construct edges in the protein graph. Hereby, I discuss two possibilities in the following subsections.

Disjunction-based Graph Construction

In this type of construction, I create an edge in the protein graph if and only if its two vertices form an edge in at least one conformation. By this definition, I have $\mathcal{PE} = \{(v_i, v_j) : v_i, v_j \in \mathcal{PV}\ s.t\ \min_{m=1,2,..,M} D_{i,j}^{(m)} \leq \delta\ \}$, where δ is an edge cutoff threshold in Å. An edge's weight is the number of conformations where its distance is less than δ. Formally, the weight of an edge between vertices v_i and v_j constructed through this method is defined as:

$$w_{i,j} = \sum_{m=1}^{M} \mathbb{1}((D_{i,j}^{(m)} \leq \delta) \tag{4.1.2}$$

where $\mathbb{1}(x)$ is an indicator whose value is one if x is true or zero otherwise.

Conjunction-based Graph Construction

In the conjunction-based approach, I construct an edge from two vertices in the protein graph if and only if those two vertices form an edge in every conformation. Thus, I have $\mathcal{PE} = \{(v_i, v_j) : v_i, v_j \in \mathcal{PV}\ s.t\ \max_{m=1,2,..,M} D_{i,j}^{(m)} \leq \delta\ \}$. Their weights are calculated through reciprocally exponential variance over all conformations such that their values approach one if their corresponding edges' variability are low. On the other hand, the high-variance edges have small weights. The mathematical definition of a weight between v_i and v_j is the following:

$$w_{i,j} = \exp\left(-\frac{\sum_{m=1}^{M}(D_{i,j}^{(m)} - \overline{D}_{i,j})^2}{K-1}\right) \tag{4.1.3}$$

where $\overline{D}_{i,j}$ is the empirical average over $D_{i,j}^{(m)}$.

In Discussion section 6.1.1, I assessed the choices of these protein graph constructions in term of error of inconsistency and found that the conjunction-based graph construction is

superior to the other. Thus, I only consider the later construction into the overall algorithm.

Rigidity Definition

I quantify the rigidity between two structures via RMSD developed by [16]. To support multiple input protein structures, I define the average RMSD over all pairs of structures. Thus, for any subset $\mathcal{S} \subseteq \mathcal{PV}$ we have

$$RMSD(\mathcal{S}) = \frac{2}{M(M-1)} \sum_{m=1}^{M-1} \sum_{m'=m+1}^{M} RMSD_{\mathcal{S}} \left(X^{(m)}, X^{(m')} \right) \tag{4.1.4}$$

where $RMSD_{\mathcal{S}} \left(X^{(m)}, X^{(m')} \right)$ is the minimal root mean square between m-th and $m^{'}$-th conformation reduced to atoms in $\mathcal{S}$.

4.1.2. Coarse Graining of the Protein Graph

Due to the thickly connected subsets of nodes in the protein graph, I reduce the size of the protein graph through Louvain algorithm [39, 75, 78] which partitions the set of vertices into communities. I determine the resolution parameter of Louvain algorithm such that the communities are (i) small enough to consist of, with a few exceptions, amino acids belonging to the same rigid domain, and (ii) big enough to make the calculation of the generalized Viterbi algorithm possible.

Suppose $\mathcal{C}$ be a partition of the protein graph, I denote its reduced graph, the coarse-grained graph as $\mathcal{CG} = (\mathcal{CV}, \mathcal{CE})$. Two communities (or vertices) in the reduced graph $c_1, c_2 \in \mathcal{C}$ are linked by an indirected edge $(c_1, c_2) \in \mathcal{CE}$ if and only if there exists a pair of amino acids $a_1 \in c_1$ & $a_2 \in c_2$ such that (a_1, a_2) is an edge in the protein graph.

I calculate the fluctuation between two communities c_1, c_2 according to the distance matrix D by the means of mean variance defined as

$$\xi_D(c_1, c_2) := \frac{1}{|c_1||c_2|(M-1)} \sum_{a_1 \in c_1} \sum_{a_2 \in c_2} \sum_{m=1}^{M} \left(D^{(m)}_{a_1,a_2} - \frac{1}{M} \sum_{m'=1}^{M} D^{(m')}_{a_1,a_2} \right)^2 \tag{4.1.5}$$

This mean variance quantifies the variability between two communities/clusters in the protein graph. Hence, its value tends to approach zero when these two communities are rigid to each other which implies they are in the identical rigid domain. Otherwise the value of mean variance will become large. When the distance matrix D is clear in the context, I would skip it to keep the notation simple.

Additionally, I also use $RMSD(\mathcal{CG})$ to denote the RMSD of the corresponding protein graph of $\mathcal{CG}$.

4.1.3. Line Graph Transformation

A line graph of an indirected graph is a transformation such that edges in the original graph become vertices in the corresponding line graph [79]. Two vertices in the line graph are linked by an edge if two corresponding edges in the original graph are incident (share a common vertex). The other two vertices of these two incident edges are called two ended vertices/nodes.

In this study, let us denote a line graph of the coarse-grained graph as $\mathcal{LG} = (\mathcal{LV}, \mathcal{LE})$ where a set of vertices $\mathcal{LV}$ coincides with the edges set in the coarse-grained graph, i.e. $\mathcal{LV} = \mathcal{CE}$. An edge links two vertices $v_1, v_2 \in \mathcal{LV}, (v_1, v_2) \in \mathcal{LE}$ if they share a common vertex and their two ended vertices are not linked in the coarse-grained graph.

Moreover, the mean variance of a vertex in the line graph $v \in \mathcal{LV}$, denoted as $\xi(v)$, is calculated by applying the Equation (4.1.5) to its corresponding two vertices in the reduced graph. Similarly, the mean variance of an edge in the line graph $\xi(e), e \in \mathcal{LE}$ is this mean variance calculation applying to its corresponding two ended vertices in the coarse-grained graph.

4.1.4. Applying the Outlier Detection

From a line graph derived from a coarse-grained graph, I observed that the bigger the mean variance of a vertex in the line graph is, the more likely its two corresponding vertices in the coarse-grained graph belong to different domains. Likewise, the two ended vertices of an edge in the line graph tended to belong to two domains if the mean variance of this edge is high. However, it was not trivial to obtain that mapping which is independently valid regardless to proteins.

Let us denote a vertex in the line graph be an inter vertex if its two corresponding vertices in the coarse-grained graph belong to different domains, or be an intra vertex otherwise. Similarly, an edge in the line graph was denoted as an inter edge/intra edge if its two corresponding ended nodes belong to the different or identical domains, respectively.

I noted that the mean variance of inter and intra vertices/edges follow two different but overlapped distributions which could be estimated through expectation maximization procedure. Nonetheless, I obtained an unsatisfactory result due to the inadequate number of inter vertices/edges. Thus, I assumed that the mean variance of intra vertices/edges followed a distribution and the values of inter vertices/edges were outliers.

I applied the outliers detection mentioned in the Section 3.5 with the default cutoff value 3.5. Suppose there is a line graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N vertices $\mathcal{V}$ and M edges $\mathcal{E}$. Without loss of generality, I denote $\mathcal{A}_{vertex} = [\xi(v_1), \xi(v_2), .., \xi(v_N)]$ be an ascending array consisting of mean variance of vertices in the line graph $\mathcal{G}$. Similarly, $\mathcal{A}_{edge} = [\xi(e_1), \xi(e_2), .., \xi(e_M)]$ is an ascending array of mean variance of edges in $\mathcal{G}$. I introduce an outlier indicator γ (though

I used the same notation, this γ is not the parameter mentioned in CPM Section 3.4) which is

$$\gamma(v|\mathcal{A}_{vertex}) = \gamma_v := \begin{cases} -1 & \text{if } v \text{ is outlier in } \mathcal{A}_{vertex}; \\ +1 & \text{otherwise.} \end{cases}$$

and

$$\gamma\big(e|\mathcal{A}_{edge}\big) = \gamma_e := \begin{cases} -1 & \text{if } e \text{ is outlier in } \mathcal{A}_{edge}; \\ +1 & \text{otherwise.} \end{cases}$$

When the arrays of mean variance of vertices and edges were unambiguous in the context, I omitted these arrays and put the notations of vertex or edge into subscript. In this study, the outliers are always the ones whose mean variance are larger than values of non-outliers. It is also possible to relax the outlier indicator by expanding the outliers set toward the non-outliers whose values are biggest.

4.1.5. Applying the Generalized Viterbi Algorithm to the Labels Inference

The difficulty of initializing parameters such as the number of rigid domains is one of the main shortcomings of several existing methods [14, 18, 19]. I solve this issue by proposing another method where I represent a protein with different conformational structures as a protein graph where I can reduce the graph size by clustering algorithms such as Louvain. The fundamental idea is to use the generalized Viterbi algorithm to infer a binary labeling indicating if a pair of communities in the reduced graph belong to the identical or different domains. To have such binary labeling on the vertices in the reduced graph, I ran the Viterbi algorithm on the line graph of the coarse-grained graph. As a result, I obtained a labeling on vertices in the line graph, which is equivalent to the labels of edges in the coarse-grained graph. By convention, in the line graph, I label a vertex "-1" if it is an inter vertex, or "$+1$" if it is an intra vertex.

Let us consider a line graph derived from the coarse-grained graph as the site graph in CRF notations (Section 3.2.2). The quality function or unnormalized probability of the site graph from Equation (3.2.4) could be rewritten as

$$p(y|\mathcal{V},\mathcal{E}) \sim \exp\left(\sum_{v\in\mathcal{V}} \Psi^{(1)}(v,y_v) + \sum_{(v_1,v_2)\in\mathcal{E}} \Psi^{(2)}(v_1,v_2,y_{v_1},y_{v_2})\right) \tag{4.1.6}$$

My goal is to design the feature functions on vertices ($\Psi^{(1)}$) and on edges ($\Psi^{(2)}$) such that a labeling which yields the highest score is also the labeling of the true rigid domains.

I define a feature function on a vertex v along with its label y_v in such a manner that its

label would agree with its rigidity signal

$$\Psi^{(1)}(v, y_v) = \gamma_v y_v \tag{4.1.7}$$

This feature function would reward the overall quality function when a vertex label agrees with its outlier indicator's value, or penalizes otherwise. That means it is preferable to assign -1 to a vertex v if its mean variance was significantly larger than others, or $+1$ otherwise.

For every edge $e = (v_1, v_2) \in \mathcal{E}$, I defined a feature function on edges along with its labels $y_e = (y_{v_1}, y_{v_2})$ as $\Psi^{(2)}(e, y_e)$ in three different cases:

Case One: "Two values among $\gamma_e, \gamma_{v_1}, \gamma_{v_2}$ are -1" The agreements between the predicted labels of vertices and their outlier indicators would be rewarded, or penalized otherwise:

$$\Psi^{(2)}(e, y_e) := \begin{cases} +1 & \text{if } y_{v_1}\gamma_{v_1} + y_{v_2}\gamma_{v_2} = 2; \\ -1 & \text{otherwise.} \end{cases} \tag{4.1.8}$$

Case Two: "$\gamma_{v_1} = \gamma_{v_2} = +1$" This seems three corresponding vertices of v_1 and v_2 (a common vertex and two ended vertices in the coarse-grained graph) are a part of an identical rigid domain. However, the common vertex might belong to the hinge part between two rigid bodies which is likely to occur if $y_e = -1$. If this was the case, one could decide of which rigid body this common vertex is part via the comparison between $\xi(v_1)$ and $\xi(v_2)$. Thus, the feature function on e becomes:

$$\Psi^{(2)}(e, y_e) := \begin{cases} +1 & \text{if } y_{v_1} = -1, y_{v_2} = +1, \gamma_e = -1 \text{ and } \xi_{v_1} > \xi_{v_2}; \\ +1 & \text{if } y_{v_1} = +1, y_{v_2} = -1, \gamma_e = -1 \text{ and } \xi_{v_1} < \xi_{v_2}; \\ +1 & \text{if } y_{v_1} = y_{v_2} = +1 \text{ and } \gamma_e = +1; \\ 0 & \text{if } y_{v_1} y_{v_2} = -1, \gamma_e = -1 \text{ and } \xi_{v_1} = \xi_{v_2}; \\ -1 & \text{otherwise.} \end{cases} \tag{4.1.9}$$

Case Three: "other combinations of γ_e, γ_{v_1} and γ_{v_1}" The feature function on edge e becomes:

$$\Psi^{(2)}(e, y_e) := 0 \tag{4.1.10}$$

Overall, for any assigned labeling of vertices in the line graph, one could calculate its quality function via Equations (4.1.8, 4.1.9, 4.1.10). Those values are later the inputs for the generalized Viterbi algorithm to find the most probable labeling.

4.1.6. Pseudo Code

Algorithm 4.1 Pseudo code of my graph-based method to calculate rigid domains in proteins.

$\theta \leftarrow 3.5$
$\mathcal{PG} = (\mathcal{PV}, \mathcal{PE}) \leftarrow X^{(m)}, m = 1..M$ // a protein graph $\mathcal{PG}$ is constructed from M structural conformations.
$\mathcal{CG} = (\mathcal{CV}, \mathcal{CE}) \leftarrow \mathcal{PG}$ // A coarse-grained graph $\mathcal{CG}$ calculated from $\mathcal{PG}$ through the Louvain algorithm.
$\boldsymbol{function}$: $Segment(\mathcal{CG})$
Final_List $\leftarrow$ [] // An empty list containing disconnected subgraphs
if RMSD$\left(\mathcal{CG}\right) > \theta$ **then**
    $\mathcal{LG} = (\mathcal{LV}, \mathcal{LE}) \leftarrow \mathcal{CG}$ // A line graph $\mathcal{LG}$ is constructed from the coarse-grained graph.
    $y_{\mathcal{LG}} \leftarrow Viterbi(\mathcal{LG})$ // Using the generalized Viterbi algorithm to calculate the most probable labels of the line graph $\mathcal{LG}$
    $y_{e:\, e \in \mathcal{CE}} \leftarrow y_{\mathcal{LG}}$ // Tracing back labels of edges in the coarse-grained graph from the labels of vertices in the line graph
    $[\mathcal{CG}_1, .., \mathcal{CG}_K] \leftarrow y_{e:\, e \in \mathcal{CE}}$ // Obtaining K disconnected subgraphs by removing negative labels of edges in the coarse-grained graph $\mathcal{CG}$
    if $K > 1$ **then**
        for all $g \in [\mathcal{CG}_1, .., \mathcal{CG}_K]$ **do**
            if RMSD$\left(g\right) > \theta$ **then**
                Run **Segment**(g)
            else
                $Final\_List$.add(g)
            end if
        end for
    else
        if $\mathcal{CG}_1 \neq \mathcal{CG}$ **then**
            Run **Segment**($\mathcal{CG}_1$)
        else
            Relaxing the outliers detection by enlarging the outliers set toward the top 5% biggest values of the non-outliers set.
            Run **Segment**($\mathcal{CG}$) with this new relaxation setting.
        end if
    end if
end if
return $Final\_List$
$\boldsymbol{end\ function}$

The above algorithm is not guaranteed to converge, yet when applying to most of my ex-

periments, the algorithm ended in a few iterations. In the implementation, I use a maximum number of iteration parameter (typically 10) which determines when to break out the loop. Overall, this algorithm gave us a list of sub-graphs which is a partition of the coarse-grained graph.

4.1.7. Finalizing the rigid domain segmentation

After obtaining a list of sub-graphs of the coarse-grained graph, I then trace back to the associated protein graph. Consequently, I obtain a mutually exclusive partition of the protein graph, denoted as $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, .., \mathcal{S}_L\}$. The merging algorithm does the following:

Merging Algorithm

$val \leftarrow \max_{\mathcal{S}_i, \mathcal{S}_j \in \mathcal{S}} \frac{\text{RMSD}\left(\mathcal{S}_i\right) + \text{RMSD}\left(\mathcal{S}_j\right)}{\text{RMSD}\left(\mathcal{S}_i \cup \mathcal{S}_j\right)}$

while $val > 1$ **do**

$\{\mathcal{D}_*, \mathcal{D}'_*\} \leftarrow \arg\max_{\{\mathcal{D}, \mathcal{D}'\} \in \mathcal{S}} \frac{\text{RMSD}\left(\mathcal{D}\right) + \text{RMSD}\left(\mathcal{D}'\right)}{\text{RMSD}\left(\mathcal{D} \cup \mathcal{D}'\right)}$

$\mathcal{D}_* \leftarrow \mathcal{D}_* \cup \mathcal{D}'_*$

Remove $\mathcal{D}'_*$ from $\mathcal{S}$

$val \leftarrow \max_{\{\mathcal{D}, \mathcal{D}'\} \in \mathcal{S}} \frac{\text{RMSD}\left(\mathcal{D}\right) + \text{RMSD}\left(\mathcal{D}'\right)}{\text{RMSD}\left(\mathcal{D} \cup \mathcal{D}'\right)}$

4.2. Identification of DNA-Binding Sites in Proteins Using Jensen–Shannon Divergence

In the following sections, I present the materials and methods I use to devise a new residue-wise feature to predict DNA-binding sites in proteins. The contents and terminologies of this section follow our publication [25] (see Appendix A.2).

4.2.1. Materials

In order to train and validate a new feature, I made a training set which consists of 263 protein-DNA complexes and associated MSAs. Those complexes were the selection from DBP-374 data set published by Wu et al. [27] where I removed any protein whose MSA has less than 125 rows. I then created the MSAs through HHblits with the UniProt20 database (version from June 2015) [42].

Following the line from [27], I classified a residue in a protein as DNA-binding site if the Euclidean distance between any atom of this residue and any atom of DNA molecule in the

DNA-protein complex was less than 3.5 or 5.0Å; or non-binding site otherwise. From the 263 DNA-protein complexes with cutoff distance 3.5Å, my set comprised 4298 binding sites and 44805 non-binding sites. With the cutoff distance 5.0Å, I obtained 7211 binding sites and 41892 non-binding sites. I validated my methods through five-fold cross validation.

4.2.2. Methods

Jensen–Shannon divergence (JSD, explained in Section 3.1.5) is a useful method to extract a new feature. Grosse et al. [80] used it to distinguish signals from two (or more) sources. Capra and Singh [61] carefully examined the effectiveness of several information theory-based measures such as Shannon entropy, von Neumann entropy, relative entropy, and sum-of-pair measures to evaluate the sequence conservation and figured out JSD was the superior candidate for that context. Gültas et al. [59] also utilized JSD in context of quantum information theory to find the coupled mutation in proteins.

Given a protein under study, let M be its MSA whose first row is the sequence of amino acids of this protein. Every k-th column in a matrix M is encoded by a 20×20 *matrix* $C(M_{\bullet k})$ whose rows and columns were indexed by 20 amino acids. For every ordered pair of amino acids (a, a'), the value of matrix coefficient $C(M_{\bullet k})_{a,a'}$ is the number of ordered pairs (i, j) such that $M_{ik} = a$, $M_{jk} = a'$, and $i \neq j$.

A null background distribution, denoted as p_{nd}, encodes a distribution of pairs of amino acids on the non-binding sites. The calculation of p_{nd} was as follows. For every protein and its associated multiple sequence alignment M in the training data set, I denoted a set of non-binding site positions as $\mathcal{NB}$. For every position $k \in \mathcal{NB}$, I computed its feature matrix $C(M_{\bullet k})$. Then, I calculated C_{nd} which is the summation over all matrices $C(M_{\bullet k})$, where M ranged over all training MSAs and k ranged over associated non-binding sites set $\mathcal{NB}$. After that, each row of the matrix C_{nd} were added up, resulting to a vector which was later normalized to obtain p_{nd}.

Given a protein and its MSA, I denoted p_k is an empirical distribution of pairs of amino acids in k-th column of its MSA. The calculation of p_k is in similar manner as the calculation of p_{nd} without the summation all over a protein set and only at a specific k-th column.

From the training data set, I observed that there was a significant difference between $JSD(p_{\{B\}}, p_{nd})$ and $JSD(p_{\{NB\}}, p_{nd})$ where $\{B\}$ and $\{NB\}$ were sets of indices of binding sites and non-binding sites in a protein, respectively. Thus, the main idea is to design a new sequence-based feature to predict DNA-binding sites in proteins which takes advantage of *Jensen–Shannon divergence*.

To extract a new feature from a residue in a protein, I utilized a sliding window method mentioned in [61] where window size is $2n+1$, with $n \in \mathbb{N}$ is the window radius. Figure 4.1 shows how sliding windows were applied in this study. For every k-th column of the MSA matrix M, I denoted $\Lambda(k)$ be a set of indices of a window surrounding k, thus $\Lambda(k) =$

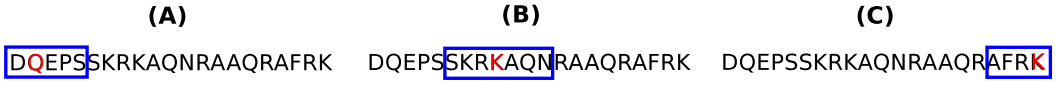

Figure 4.1.: Sliding windows with radius $n = 3$ in three scenarios. The target residues are colored in red and their corresponding sliding windows in blue. (A) When the target residue is close to the beginning of the sequence, the left wing of the window is partially missed. (B) A full sliding window of a target residue. (C) A sliding window of an ended residue, resulting in missing its right wing.

$\{k-n, k-n+1, .., k-1, k, k+1, .., k+n-1, k+n\} \bigcap \{1, 2, .., L\}$ where L is the length of the protein sequence. I defined the score at the k-th column of the matrix M as the weighted sum over all JSDs between p_k and p_{nd}, normalized by the sum of weights

$$\begin{aligned} \mathcal{S}_M(k) &= \frac{\sum_{i \in \Lambda(k)} (n+1-|k-i|) JSD(p_k, p_{nd})}{\sum_{i \in \Lambda(k)} (n+1-|k-i|)} \\ &= \frac{\sum_{i \in \Lambda(k)} (n+1-|k-i|) JSD(p_k, p_{nd})}{(n+1)^2 - \binom{2(n+1)-|\Lambda(k)|}{2}} \end{aligned} \tag{4.2.1}$$

where the binomial coefficient $\binom{x}{2} = x(x-1)/2$ and $|\Lambda(k)|$ is the cardinality of $\Lambda(k)$.

The above formula embedded information from spatial neighbors which was shown useful to predict a residue as functionally important [81]. The usage of sliding windows to calculate the score has been applied in [60, 61] and shown positive results. The weighting pattern for windows was inspired by Janda et al. [82] where I concentrated the biggest weight to the target residue and linearly reduced weights of other residues in the window upon to their index distances to the target residue.

The choice of the window radius should not be too big or too small. A big radius could result in indistinguishable signals between binding and non-binding windows. In contrast, a too small radius may negatively affect the performance due to lack of neighborhood information. From the suggestions of [61, 82] and from my own experiments, I noticed that sliding windows with radius $n = 3$ obtained the best results.

The window scores associated with each amino acid calculated in Equation (4.2.1) were highly dependent on the protein MSA. Thus, they were not comparable across proteins. To eliminate such MSA-dependence, I transferred the window scores via percentile procedure such as

$$f_M(k) = \frac{\left|\left\{k' | 1 \le k' \le L,\ \mathcal{S}_M(k) \ge \mathcal{S}_M(k')\right\}\right|}{L}. \tag{4.2.2}$$

This transformation is basically the determination of the percentage of scores below the current one at position k [25].

To evaluate the effectiveness of the new sequence-based feature, I annexed it to existing features such as PSSM, OBV, and SS mentioned in [27] and discussed in Section 2.3. The new feature along with existing ones would be plugged into random forest (RF) classifier for training and evaluating. In this study, I used a RF implementation in WEKA data mining software [83].

Usually, the number of non-binding sites in a protein significantly outnumbers the number of binding sites. This created an imbalance between positive and negative samples which was cumbersome for the training. In order to reduce the effect of this imbalance, I applied the bagging technique as follows. In a training phrase, I ran eleven sub-phrases where I constructed a RF classifier from a training samples set consisting of all positive samples and as twice negative samples as the positive ones, randomly chosen with replacement in the training samples pool. For each instance in the validation, the result was the majority vote of above eleven RF classifiers.

5. Results

Follows in this section I would like to present the outcomes resulting from approaches discussed in the Methods section (Section 4). This section is organized as the following. Firstly, I investigate the identification of rigid domains in proteins using graph-based method. Secondly, I present the results of identification of DNA-binding sites in proteins inferred by Random Forest classifiers utilizing Jensen-Shannon divergence as its feature.

5.1. Identification of Rigid Domains in Proteins using Graph-based Model

I evaluated my method on three different data sets. First of all, I ran my experiments on Adenylate Kinase (ADK) protein and described how the workflow proceeded. Secondly, I assessed this method on large Dyndom data set comprising of 478 non-redundant proteins. Finally, I illustrated the comparison between my method and other existing ones on various structural change proteins.

The content of this section is in our published paper in BMC bioinformatics [13] (see Appendix A.1).

5.1.1. Rigid domains of Adenylate Kinase

I first ran my methods on ADK protein whose many different conformations were available [84]. This protein is a catalyst assisting the inter-conversion of ADP[2] to ATP[3] and AMP[4]. ADK plays an essential role in the creation and storage of chemical energy molecular which is the key element for cells to grow substantially. For example, the deficiency of ADK links to moderate to severe hemolytic anemia [85]. In a closed conformation, the NMP-bidning and LID domains which attach onto the core domain of ADK facilitate the chemical reaction converting ATP and AMP into two ADP molecules. I investigated the open and closed conformations of ADK (PDB codes: 4AKE and 1AKE, both chain A). ADK protein comprises of 214 amino acids which are represented as 214 vertices in a protein graph. To

[2] Adenosine diphosphate
[3] Adenosine triphosphate
[4] Adenosine monophosphate

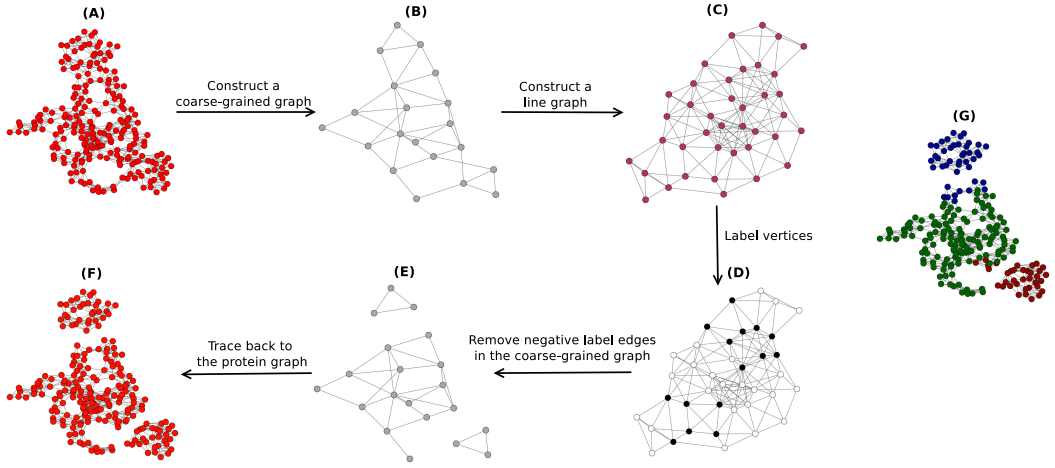

Figure 5.1.: Graph-based segmentation of ADK into rigid domains. (A) A protein graph constructed from open and closed ADK conformations. (B) a reduced/coarse-grained graph obtained by coarse graining the protein graph. (C) A line graph of the reduced graph. (D) The line graph with binary vertex labels (black: -1, white: +1) obtained via the generalized Viterbi algorithm. (E) The injective relation between edges of the reduced graph and vertices of the line graph allows us to also label the edges of the reduced graph. Edges with negative labels are removed, resulting to three disconnected subgraphs. (F) A segmented protein graph derived from disconnected subgraphs in the reduced graph. (G) ADK graph with domain annotation from literature encoded by colors.

define the neighborhood among vertices, I used the cutoff value $\sigma = 7.5$Å as a threshold.
ADK's segmentation in Figure 5.1 summarizes the workflow as the following:

Step 1: I construct a protein graph of ADK where each amino acid is a vertex in a graph (Section 4.1.1), resulting to a neighborhood graph with 214 vertices as shown in Panel 5.1.A.

Step 2: Vertices in the protein graph are grouped together through Louvain algorithm to create the coarse-grained version of the protein graph, named coarse-grained graph (Panel 5.1.B).

Step 3: I calculate a line graph (Panel 5.1.C) from the reduced graph (Section 4.1.3) as well as compute its scoring function (Section 4.1.5).

Step 4: Consequently, I apply the generalized Viterbi algorithm to estimate the most probable labels of vertices in the line graph (Panel 5.1.D).

Step 5: Once the labels of vertices in the line graph are calculated, the labels of edges in the coarse-grained graph are also implied correspondingly due to the mutual

transformation between the coarse-grained graph and the line graph. Afterward, I remove the edges with negative labels, resulting to three disconnected sub-graphs of the coarse-grained graph (Panel 5.1.E).

Step 6: In each disconnected sub-graph, I trace back to their corresponding protein sub-graphs. The resulting graph is the protein graph which is segmented into three rigid domains (Panel 5.1.F).

Step 7: I assess my segmentation by the referenced protein graph whose rigid domains was found from literature and vertices in different domains encoded by different colors (Panel 5.1.G).

As shown in Figure 5.1, my segmentation differs from the literature annotation merely at the hinges between two rigid bodies. The disagreement is mostly due to the vague membership of amino acids resided at the hinges which are incorrectly blended with other amino acids from different domains in the coarse graining.

My graph-based algorithm supports users to integrate their own prior knowledge, thus probably resulting to the improvement. For instance, my above segmentation on ADK produces, according to the literature annotation, fifteen misclassified amino acids of NMP-binding and LID domains to the core domain. Supposed I was given a segmentation of ADK from another method such as Spectrus [18] with $K = 4$ (K is the number of rigid domains). I am able to integrate this prior segmentation (or prior label) to the algorithm as follows. I shrank edge weights in the protein graph by a factor $\alpha < 1$ (typically 0.75) if their two vertices belonged to different domains according to the prior segmentation. This selectively shrinking process helped to reduce the error of inconsistency arisen in the coarse graining (discussed in the Discussion section) and thus led to the improvement of the segmentation. I ran the algorithm with the prior segmentation from Spectrus on ADK and observed that it missclassified only five amino acids of LID domain to the core domain. This demonstrated that one could significantly improve the segmentation even with some imperfect prior knowledge.

5.1.2. Rigid Segmentation Benchmark

Following the line of Nguyen et al. [19], I assessed my graph-based method on the big benchmark DynDom data set [86] which is the collection of proteins with two different conformational states. To remove the redundancy, I filtered out all proteins whose average TM-score [55] with other proteins was greater or equal 0.5, which was a threshold according to Zhang et al. [87] indicated that these proteins would have similar structure. Moreover, in the scope of this study, I only investigated into medium to large conformational changes. Thus, I also removed protein structure pairs whose RMSDs are less than 5.0 Å. As a result, I obtained 487 proteins for the assessment. Additionally, I removed domains that consisted of less than ten amino acids.

For the numeric evaluation, I utilized the two metrics which are error and overlap defined in [19]. In the following, I explain and give an example to demonstrate how these two metrics work.

Error

The error counts how often two segmentations disagree on if their pair of amino acids are on the same domain. For instance, let us consider a protein comprised of ten amino acids with two different segmentations described in the Table 5.1.

Table 5.1.: An example of a protein with two segmentations.

1^{st} segmentation	0 0 0 1 1 1 2 2 0 0
2^{nd} segmentation	1 1 2 2 2 0 0 3 3 3

The first segmentation has three domains while the second result has four ones. To present the inter-intra relationship over every pair of amino acids in a segmentation, I utilized the 10×10 square matrix (Tables 5.2 and 5.3).

Table 5.2.: Inter-intra relationships over all pair of amino acids in the 1^{st} segmentation.

	0	0	0	1	1	1	2	2	0	0
0		0	0	1	1	1	1	1	0	0
0			0	1	1	1	1	1	0	0
0				1	1	1	1	1	0	0
1					0	0	1	1	1	1
1						0	1	1	1	1
1							1	1	1	1
2								0	1	1
2									1	1
0										0
0										

Table 5.3.: Inter-intra relationships over all pair of amino acids in the 2^{nd} segmentation.

	1	1	2	2	2	0	0	3	3	3
1		0	1	1	1	1	1	1	1	1
1			1	1	1	1	1	1	1	1
2				0	0	1	1	1	1	1
2					0	1	1	1	1	1
2						1	1	1	1	1
0							0	1	1	1
0								1	1	1
3									0	0
3										0
3										

Because of the symmetry, I only need the upper triangle tables. The diagonal of those tables are always zero, thus not interesting to us. An entry value of a table would be "0" if its two corresponding vertices, according to the segmentation, belong to an identical domain, or be "1" otherwise.

The error between these two segmentations is calculated by counting the disagreements between those above tables, normalized by the number of pair of amino acids. In the other words, the disagreement is calculated by Not-Or-Exclusive operation as follows: $\frac{2\times sum(Table\ 5.4)}{10\times 9} = \frac{2\times 16}{10\times 9} = 0.356$.

Table 5.4.: The table contains disagreements between Tables 5.2 and 5.3.

	0	1	0	0	0	0	0	1	1
		1	0	0	0	0	0	1	1
			1	1	0	0	0	1	1
				0	1	0	0	0	0
					1	0	0	0	0
						1	0	0	0
							1	0	0
								1	1
									0

Overlap

The overlap takes into account the matches between two segmentations after arrangement them in a manner that they obtain the maximum agreement which is obtained by solving a low-dimensional linear assignment problem. Let us consider the same above example to illustrate how to calculate the overlap between these two segmentations. Firstly, we compute the overlap matrix which is:

Table 5.5.: The overlap table between two segmentations.

Segmentation 1 \ Segmentation 2	0	1	2	3
0	0	3	2	2
1	1	0	2	0
2	1	0	0	1

The first column and first row in the Table 5.5 contain the domain indexes of the first and second segmentation respectively. Consider the first segmentation as a reference, for each domain index we count how many times amino acids in this domain agree to other domains in the second segmentation. The highlighted number in each row is the maximum agreement it can get. Thus, the overlap between those segmentations is the sum of all maximal agreements divided by the length of the sequence, which is $\frac{(3+2+1)}{10} = 0.60$.

Though the metrics of error and overlap are calculated differently, they are highly countercorrelated.

Benchmark Assessment

Figure 5.2 shows the histogram of the error and overlap between mine and DynDom's segmentations on 487 entries in DynDom data set with the edge cutoff value 7.5 Å. The median error and overlap are 0.038 and 0.972 respectively. Particularly, around 30% of my labelings highly agree with annotation provided by DynDom (overlap ≥ 0.99). Yet, occasionally my method was unable to calculate reasonable segmentations due to two possible reasons. Firstly, the coarse-graining step failed to produce homogeneity communities, i.e., most of amino acids in a community belonged to a same domain. Secondly, the mean variance-based signals calculated from inter and intra vertices/edges in the line graph were indistinguishable. This created confusions to the scoring function and thus the most probable label did not coincide with the actual one.

I investigated examples whose segmentations according to my method disagreed with DynDom. I observed that sometimes my method suggested a more reasonable labeling

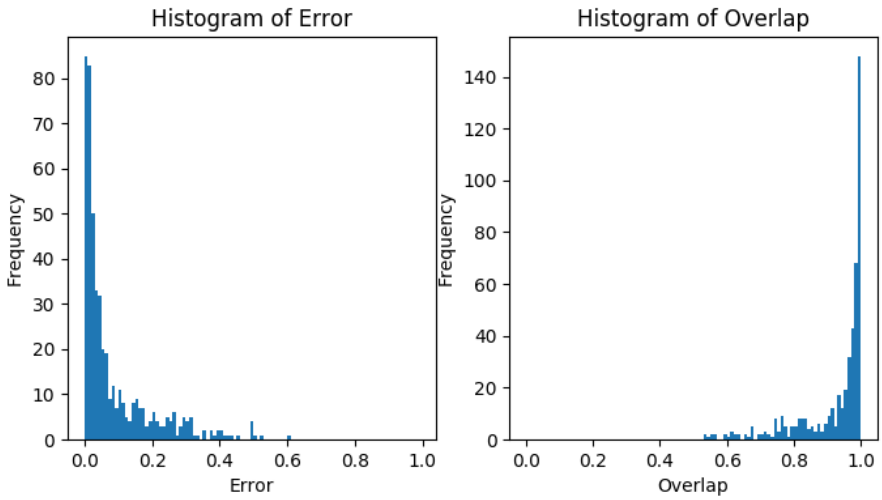

Figure 5.2.: The histogram of the error and overlap evaluated on 487 proteins in the DynDom data set.

than annotations from DynDom. For example, let us consider a *human importin subunit beta-1* protein which is an entry in DynDom data set. Panel (B) in Figure 5.3 represents my algorithm's result, run on the open and closed states of this protein (PDB code 3lww, chain *A&C*). My method produced two separate rigid domains whose RMSDs were 2.228 and 1.003Å. On the other hand, DynDom annotation suggested three rigid domains whose RMSDs were 6.843, 4.321, and 2.106Å(Panel (A) in Figure 5.3). It is noticeable that the first domain of DynDom annotation (dark green) is small, fragmented and have a relatively large RMSD. In addition, the second domain (dark red) has a significant portion which is intertwined with the third domain (dark blue). Overall, my segmentation on a *human importin subunit beta-1* protein seems more reasonable than one from DynDom according to RMSD metric as well as pictorial presentations.

I also studied the influence of the edge cutoff used in the construction of protein graphs through experiments on varying cutoff values. I summarize the results (Table 5.6) reporting the mean and median of error and overlap on 487 proteins in DynDom data set attained with various edge cutoff values. The overlap seems to be unaffected by the choice of edge cutoff, meanwhile the error is slightly dropped with bigger cutoffs. I suggest two probable explanations. Firstly, a big edge cutoff produces a denser protein graph which seemed to obtain a better coarse-grained graph (see Discussion section 6.1.1). Secondly, a denser protein graph eventually results in a denser coarse-grained graph which seems to enhance the mean-variance driven signals for the scoring function in the line graph. Nonetheless, I restricted the cutoff to smaller values due to the computational cost of the generalized

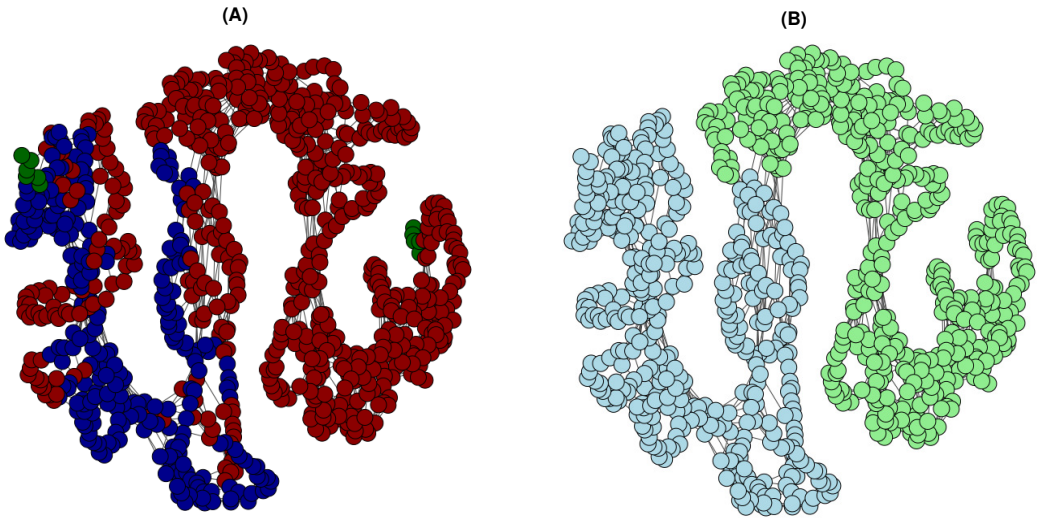

Figure 5.3.: Protein graph of human importin subunit beta-1 protein. (**A**) Segmentation suggested by DynDom: three rigid domains colored by dark green, red and blue. (**B**) My segmentation: two rigid domains colored by light green and blue.

Table 5.6.: The performance of my graph-based method with varying edge cutoff values assessed on DynDom data set.

Metric / Cutoff	Median overlap	Mean overlap	Median error	Mean error
7.5Å	0.972	0.924	0.038	0.086
10.5Å	0.977	0.924	0.034	0.083
13.5Å	0.972	0.926	0.033	0.081

Viterbi algorithm.

5.1.3. Rigid segmentation on various structural transitions

I evaluated my graph-based rigid domains estimator on various proteins studied by [19]. These proteins dynamic included different types and scales of conformational change. The PDB codes as well as other information of these proteins are summarized in the Table 5.7. Moreover, Figure 5.4 shows my segmentations along with the other methods segmentations.

Pyruvate phosphate dikinase (PPDK) is a big complex catalyzing the reversible conversion of PEP, AMP, and Pi to pyruvate and ATP [88]. I ran the method on two conformations

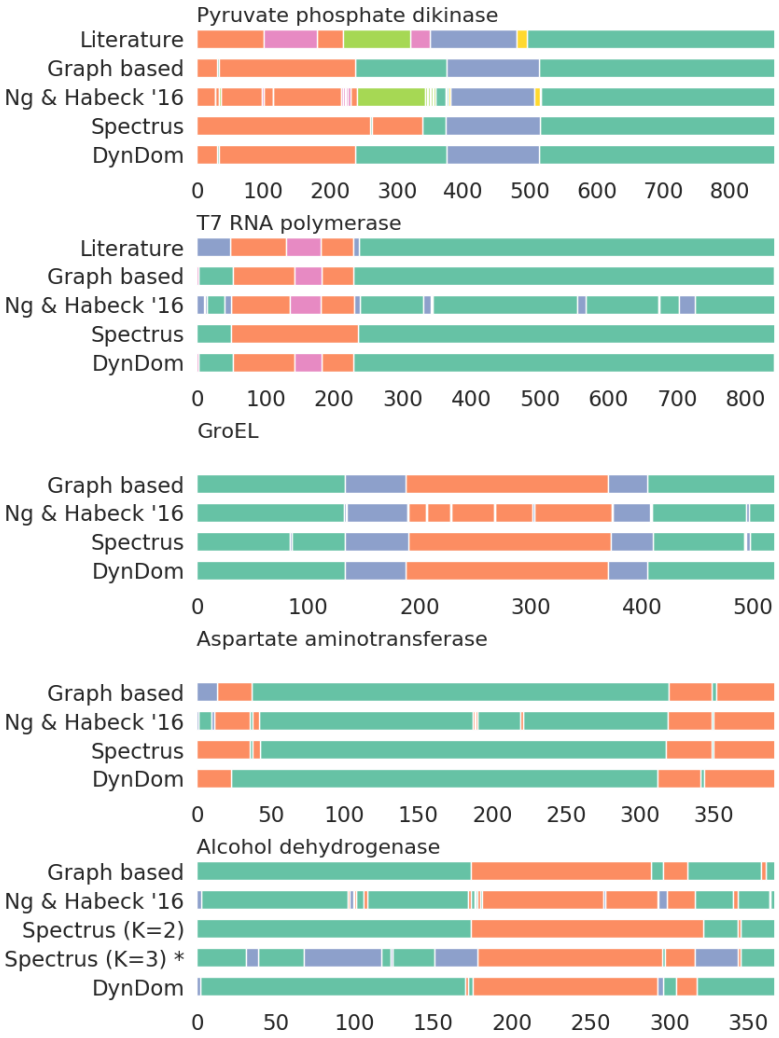

Figure 5.4.: Analysis of several proteins undergoing conformational changes on a variety of scales. Large-scale conformational changes: pyruvate phosphate dikinase, T7 RNA polymerase, GroEL. Medium-scale conformational changes: Aspartate aminotransferase, Alcohol dehydrogenase. For each protein, the segmentation found by different methods and in the literature are shown. Same color means same domain.

Table 5.7.: Proteins in different scale conformational changes involved in the assessment.

Protein	PDB code	chain ID	size
PPDK	1kc7	A	872
	2r82	A	
T7 RNA polymerase	1qln	A	842
	1msw	D	
GroEL	1aon	A	524
	1aon	H	
Aspatate aminotransferase	9aat	A	401
	1ama	A	
Alcohol dehydrogenase	1adg	A	374
	2ohx	A	

of PPDK and compared the result to the segmentation from the literature [88] as well as to the results from other methods such as Nguyen&Habeck'16, Spectrus and DynDom. As illustrated in a first panel of the Figure 5.4, my result strongly agrees with the segmentation produced by DynDom, yet both methods miss an additional rigid domain reported by the literature and in [19]. My method typically produces fewer number of rigid domains than reported in the literature because it only takes advantage of the information of conformational changes from a few structural snapshots but no further experimental information. Spectrus with $K = 3$ strongly concurs with my segmentation, except that the first domain is bigger.

T7 RNA polymerase is an enzyme involving in the commencement and expansion of RNA transcription. My segmentation along with ones from DynDom and Nguyen & Habeck'16 strongly agree with the rigid domains reported in [89]. Spectrus, however, fails to detect a refolding loop inserted in the N-terminal domain.

Another big conformational transition, the chaperonin GroEL complex [90] is another case study. This complex provides a protected environment to help protein folding and prevent aggregation. In this case, all methods strongly agree with each other.

I also assessed my method on intermediate scale structural change. Aspartate aminotransferase (AST) is an enzyme specialized for amino acid metabolism that catalyzes the reversible transfer of an α-amino group between glutamate and aspartate [91]. Though my method identified a small additional rigid domain, its segmenting result mostly agrees with the other methods. Another medium-sized structural transition complex is Alcohol dehydrogenase (AhD) [92, 93] which is an enzyme assisting the decomposition of alcohol into aldehyde. The segmentation of my graph-based method strongly agrees with the estimation

from DynDom. Spectrus reports an additional domain when obtaining the maximum score with $K = 3$. Though Spectrus with $K = 2$ domains has a lower score value, its segmentation is more consistent to my result and to DynDom.

5.2. Identification of DNA-Binding Sites in Proteins Using Jensen–Shannon Divergence

To evaluate my new sequence-based feature, I firstly investigated the new feature along with three existing ones in the RF classifier. Secondly, to demonstrate the usefulness of the new feature, I used the trained RF to analyzes the proto-oncogenic transcription factor MYC-MAX complex(PDB-ID: 1NKP) which does not belong to the training set.

The content of this Section was published in [25], included in this thesis in Appendix A.2.

5.2.1. Cross Validation on Benchmark

I constructed an RF classifier from 4298 positive and 44,805 negative instances (cutoff distance 3.5Å) or 7211 positive and 41892 negative instances (cutoff distance 5.0Å) extracted from 263 proteins. I conducted the evaluation under five-fold cross validation process where the samples were randomly divided into five parts. I assessed the performance by iteratively considering one part as a test set and these other four parts as training set.

To construct features for a RF classifier, I combined my sequence-based feature ($\mathfrak{f}_{\mathbb{JSD}}$) with other widely used features such as $\mathfrak{f}_{\mathbb{PSSM}}$, $\mathfrak{f}_{\mathbb{OBV}}$ and $\mathfrak{f}_{\mathbb{SS}}$. As shown in Tables 5.8 and 5.9, the involving of the new feature significantly boosted the performance of the RF classifier utilized to identify DNA-binding sites in proteins. I noticed that by concatenating the new feature into the existing ones, the RF's sensitivity is substantially increased while its specificity is slightly decreased. Consequently, the other assessing metrics such as Matthews correlation coefficient (MCC) and area under curves which take every true or false predictions into account indicate the new feature provides new informative aspects.

In order to validate the positive effect of the new sequence-based feature, I further analyzed two additional data sets RBscore [33] and PreDNA [94]. RBscore and PreDNA data sets originally consist of 381 and 224 DNA-binding proteins respectively, yet I removed a few proteins because either a few of them are already included in the old data set or their multiple sequence alignments are not sufficient to retrieve useful information. Similarly to previous evaluation, I constructed an RF classifier from 263 proteins in my previous data set (the one I used in five-fold cross validation). For testing, I randomly selected 60 proteins from each data set RBscore and PreDNA respectively and the results in Tables 5.10 and 5.11 suggest that the new feature add a great complementary effect to the existing ones.

Table 5.8.: Prediction performance of RF classifier on different features using a cut-off of 3.5 Å. The prediction system was evaluated by five-fold cross validation.

Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
f_{PSSM}	0.292	0.963	0.307	0.777	0.313
$f_{PSSM} + f_{JSD}$	0.385	0.949	0.349	0.795	0.369
$f_{PSSM} + f_{SS}$	0.339	0.958	0.334	0.794	0.338
$f_{PSSM} + f_{SS} + f_{JSD}$	0.416	0.95	0.378	0.808	0.390
$f_{PSSM} + f_{OBV} + f_{SS}$	0.367	0.968	0.398	0.838	0.413
$f_{PSSM} + f_{OBV} + f_{SS} + f_{JSD}$	0.422	0.958	0.409	0.837	0.425

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

Table 5.9.: Prediction performance of Random Forest (RF) classifier on different features using a cut-off of 5.0 Å. The prediction system was evaluated by five-fold cross validation.

Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
f_{PSSM}	0.286	0.966	0.350	0.778	0.425
$f_{PSSM} + f_{JSD}$	0.395	0.95	0.407	0.801	0.487
$f_{PSSM} + f_{SS}$	0.334	0.963	0.386	0.796	0.455
$f_{PSSM} + f_{SS} + f_{JSD}$	0.424	0.951	0.436	0.814	0.513
$f_{PSSM} + f_{OBV} + f_{SS}$	0.337	0.975	0.431	0.830	0.517
$f_{PSSM} + f_{OBV} + f_{SS} + f_{JSD}$	0.419	0.958	0.450	0.832	0.535

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

Table 5.10.: Prediction performance of RF classifier on RBscore data set using different distance cut-offs.

Cutoff	Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
3.5Å	f_{PSSM}	0.458	0.974	0.476	0.866	0.460
	$f_{PSSM} + f_{JSD}$	0.560	0.965	0.514	0.894	0.518
	$f_{PSSM} + f_{SS}$	0.512	0.970	0.501	0.878	0.476
	$f_{PSSM} + f_{SS} + f_{JSD}$	0.581	0.960	0.511	0.899	0.520
	$f_{PSSM} + f_{OBV} + f_{SS}$	0.517	0.976	0.534	0.896	0.528
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{JSD}$	0.580	0.967	0.540	0.907	0.543
5.0Å	f_{PSSM}	0.445	0.977	0.528	0.873	0.589
	$f_{PSSM} + f_{JSD}$	0.553	0.968	0.579	0.899	0.643
	$f_{PSSM} + f_{SS}$	0.490	0.973	0.547	0.880	0.602
	$f_{PSSM} + f_{SS} + f_{JSD}$	0.578	0.963	0.583	0.902	0.648
	$f_{PSSM} + f_{OBV} + f_{SS}$	0.499	0.980	0.584	0.895	0.641
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{JSD}$	0.570	0.968	0.595	0.908	0.661

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

The evaluation of the new feature over three different data sets suggests that by adding the new feature, the RF classifier significantly detects more true positive, yet the number of false positive is slightly increased. In order to take both true positive and true negative into account, I evaluated the results on other metrics such as area under the receiver operating characteristics curve (AUC-ROC) or under the precision-recall curve (AUC-PR). I noticed that by adding the sequence-based feature, the RF classifier consistently achieves better results. Nevertheless, the positive contribution of the new feature is reduced if the existing features has expanded to include others.

5.2.2. Position Analysis of the MYC-MAX Protein

I further evaluated the effect of the sequence-based feature on a protein complex MYC-MAX (PDB-Entry 1NKP), a proto-oncogenic transcription factor, playing a key role in cell proliferation. This complex is believed over-expressed in many different types of cancer [95]. A core element of the promoter that consists of six nucleotides is binded by MYC-MAX transcription factors and consequently activates transcription of the underlying genes [96].

The MYC protein consists of 88 amino acids, ten of them are considered as DNA-binding sites due to the fact that their distances to DNA are less than 3.5 Å. The RF classifier with the new feature (f_{JSD}) combining to existing ones was able to predict in total seventeen residues as DNA-binding sites. Seven of them (H906, N907, E910, R913, R914, P938, K939) are the true DNA-binding sites in MYC protein. I have noticed that while the DNA-

Table 5.11.: Prediction performance of RF classifier on PreDNA dataset using different distance cut-offs.

Cutoff	Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
3.5Å	f_{PSSM}	0.378	0.977	0.410	0.840	0.391
	$f_{PSSM} + f_{JSD}$	0.498	0.963	0.448	0.865	0.453
	$f_{PSSM} + f_{SS}$	0.393	0.975	0.417	0.847	0.402
	$f_{PSSM} + f_{SS} + f_{JSD}$	0.501	0.966	0.461	0.872	0.463
	$f_{PSSM} + f_{OBV} + f_{SS}$	0.428	0.977	0.458	0.867	0.451
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{JSD}$	0.511	0.97	0.488	0.885	0.488
5.0Å	f_{PSSM}	0.373	0.979	0.463	0.833	0.496
	$f_{PSSM} + f_{JSD}$	0.485	0.962	0.495	0.858	0.540
	$f_{PSSM} + f_{SS}$	0.389	0.977	0.470	0.839	0.501
	$f_{PSSM} + f_{SS} + f_{JSD}$	0.490	0.963	0.501	0.863	0.550
	$f_{PSSM} + f_{OBV} + f_{SS}$	0.395	0.980	0.488	0.858	0.530
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{JSD}$	0.480	0.968	0.511	0.874	0.563

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

binding sites R913, R914, P938, and K939 are also detected without my new feature, the remaining three binding sites can only be detected if I include my sequence-based feature into the feature extraction.

The second protein in this complex is MAX protein folded by sequence of 83 amino acids. By expanding my new feature into feature extraction, the RF classifier was able to predict fourteen DNA-binding sites, eight of them (H207, N208, E211, R212, R214, R215, S238, R239) are true positive. It is remarkable to notice that without using my new feature, the RF only detect two true binding sites (S238, R239) out of nine. The result is illustrated and summarized via Figure 5.5 and Table 5.12.

The analysis of MYC-MAX complex suggests that including the new feature into the feature extraction significantly improve the sensitivity, and thus the MCC, while the specificity is moderately reduced. This result is consistent with the other benchmarks I analyzed above.

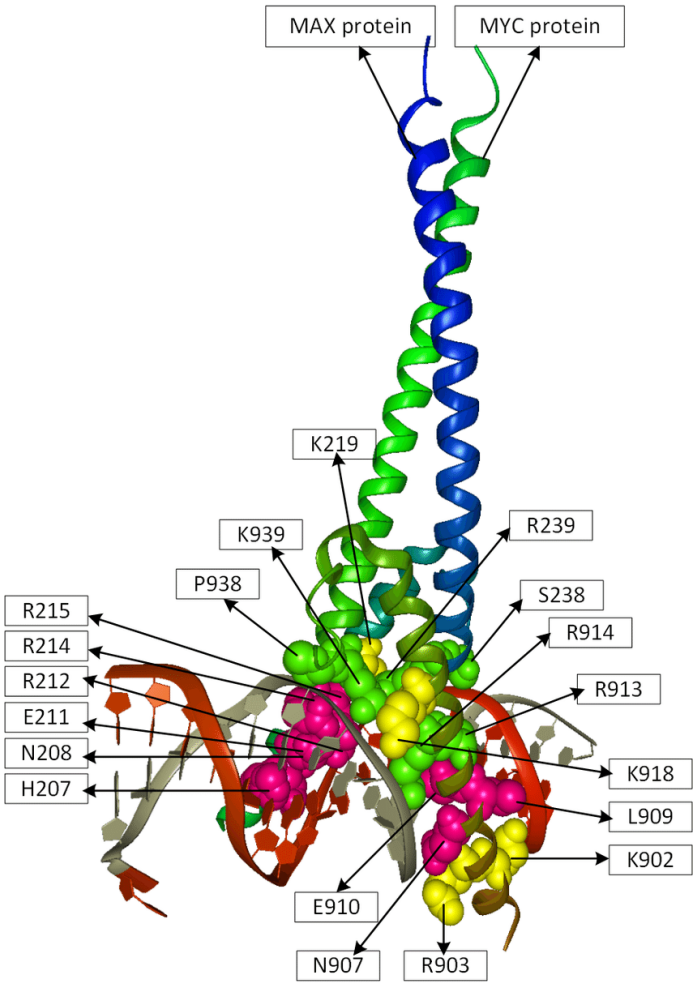

Figure 5.5.: DNA-binding sites in proto-oncobenic transcription factor MYC-MAX protein complex (PDB-Entry 1NKP). Green spheres denote positions of the DNA-binding sites in both proteins which are detected by RF classifier either using the existing features (f_{PSSM} , f_{OBV}, and f_{SS}) alone or combining my new features with these existing features together. Purple spheres show the localization of additional binding sites which were only found by RF classifier using my new features with existing features. Moreover, there are further three binding sites in MYC protein and one binding site in MAX protein, shown with yellow spheres, that could not be identified by the classifier.

Table 5.12.: Prediction performance of RF classifier on different features using a cut-off of 3.5 Å for MYC-MAX protein complex (Protein Data Bank (PDB)-Entry 1NKP).

Protein	Feature	Sensitivity	Specificity	MCC
MYC	$f_{PSSM} + f_{OBV} + f_{SS}$	0.300	0.941	0.282
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{JSD}$	0.700	0.853	0.448
MAX	$f_{PSSM} + f_{OBV} + f_{SS}$	0.222	1.0	0.447
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{JSD}$	0.888	0.906	0.664

6. Discussion

In the following section, I discuss the methods described in this thesis as well as their corresponding results. This section comprises two smaller subsections with the aims to answer the research questions arisen at the commence of this thesis. In each subsection, I deliver my insights of strengths and limitations of my methods.

6.1. Answers for Research Questions Concerning the Task of Rigid Domains in Proteins Detection

The results described in Section 5.1 have shown that my graph-based methods are successfully able to partition proteins into their rigid domains. To have a better understanding how the methods work, let us discuss their key features and the impacts of the algorithmic parameters.

The notations and the content of this discussion is based on the discussion section of our paper [13] (see Appendix A.1).

6.1.1. Coarse-graining Procedure

I studied different clustering methods and figured out that the Louvain clustering algorithms were the most suitable for my approach because they allowed me to integrate the graph structure into their calculations. In particular, I decided to choose the Louvain algorithm with Pott models [39] due to its comprehensible interpretation. With the Pott model, the clustering algorithm tried to maximize the number of internal edges within communities while keeping their sizes relatively small. In other words, the splitting of a big cluster into two smaller ones depends on the density of links between those two smaller clusters (see Section 3.4). Those interpretations of clustering fit my graph-based model where I also assumed internal connections in rigid domains are more frequent than ones between domains.

Even though the coarse-graining process on the protein graph occasionally created wrong clusters, it significantly enhanced the mean variance-driven signals for the quality function. In the following subsections, I discuss and analyze two main effects of the coarse graining which are the advents of inconsistency error and signal enhancement.

Inconsistency Error

I introduce a metric, called inconsistency error, to measure the efficiency of the protein graph construction and the coarse graining. This metric quantifies the heterogeneity of communities in the coarse-grained graph weighted by their sizes. For the formal definition, let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with a set of N vertices $\mathcal{V}$. For every i-th verex $v_i \in \mathcal{V}$, its label is denoted as σ_i. In addition, let $C = \{C_k\}$ be a partition of vertices into communities $C_k \subset \mathcal{V}$ resulted from the coarse graining. I define the inconsistency error of the coarse-graining procedure as

$$\mathcal{IE}(C|\mathcal{G}) = 2 \sum_{C_k \in C} \frac{|C_k|}{N} \frac{\sum_{i<j \in C_k} |\sigma_i \neq \sigma_j|}{|C_k|(C_k - 1)} \tag{6.1.1}$$

where $|C_k|$ is the cardinality of a community C_k and $|\sigma_i \neq \sigma_j|$ is 1 if $\sigma_i \neq \sigma_j$, or 0 otherwise. This above entity is the average number of labelling mismatches within a cluster weighted by the cluster size. The entity $\mathcal{IE}(C|\mathcal{G})$ will be zero if the coarse graining gives us all homogeneous clusters. However, the zero value of inconsistency error is not all what I like to obtain. It is important to notice that the coarse graining could easily achieve zero inconsistency by assigning each vertex as its own cluster, yet it does not produce any benefit. Thus, it is important to control the coarse graining in a manner that it produces small inconsistency while the number of the clusters is significantly smaller than the number of vertices. On the other hand, this error will approach one if the coarse graining totally gives wrong clusters. This only happens when each vertex has distinguish label, but all are grouped into a single big cluster.

I firstly study different ways to construct protein graphs from multiple conformations mentioned in Section 4.1.1. In short, in the disjunction-based protein graph construction, denoted as type (I), I created an edge between two vertices if their distance is smaller than a cutoff in *at least one* conformation. The edge weight is the number of such conformations. In contrast, in the conjunction-based protein graph construction, denoted as type (II), an edge between two vertices is created if and only if its distance is smaller than a cutoff in *all* conformation. Additionally, I assigned a weight to an edge by its reciprocal exponentiated variance computed over all conformations (see Equation 4.1.3). Such weight assignment follows the idea that low-variance edges have a weight close to one and high-variance edges are assigned to a weight close to zero.

Figure 6.1 shows that the second protein graph construction consistently outperformed the first type in term of the inconsistency error. A possible explanation is as follows. The second type of construction rule produces a sparser protein graph where an edge between two vertices are created only when ones are certain that these two are close in all cases. Consequently, rigid bodies in a protein graph tend to have more edges than the ones between bodies. Thus, the coarse-graining procedure obtains less error of inconsistency.

Additionally, I examined varying values of edge cutoff, ranging from 7.5 to 13.5Å. Ac-

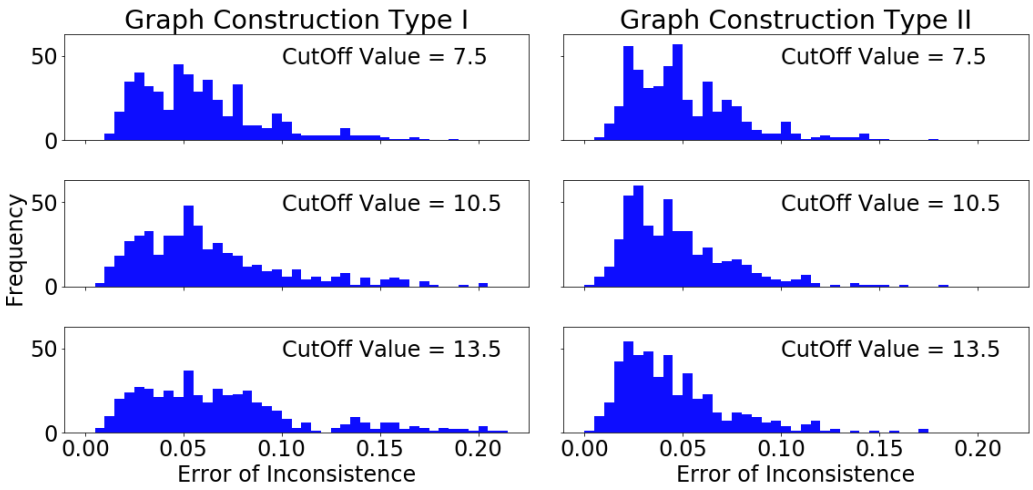

Figure 6.1.: Histogram of inconsistency error from graph construction type I and II and their varying cutoff values respectively.

cording to the results from Figure 6.1, there was a small, but not significant improvement of the inconsistency error for larger cutoff values.

Signal Enhancement

As mentioned in Section 4.1.4, the efficiency of the quality function (Equation 4.1.6) depends on the separability between the mean variance values of inter- and intra-vertices or edges in the line graph.

Given an array of mean-variance values calculated from inter- and intra-vertices in the line graph, I used the area under the ROC curve (AUC) to measure how well the mean-variance can distinguish between these two group of vertices. Likewise, to measure the separability between inter- and intra-edges in the line graph, we apply the AUC in the identical manner.

The AUCs calculated on vertices and edges in the line graph derived from the coarse-grained graph (red bars in Panels (A) and (B) of Figure 6.2) are significantly larger than the ones calculated on vertices and edges in the line graph derived directly from the protein graph (blue bars in Panels (A) and (B) of Figure 6.2). Thus, the illustrations in Figure 6.2 give us a strong evidence of the advantage of using the coarse graining in my methods.

Overall, in my study, I adjusted the resolution parameter of Louvain algorithm so as to produce about twenty clusters of medium size. Too big clusters could result in the increase of the inconsistency error because amino acids in hinge regions tend to be merged together.

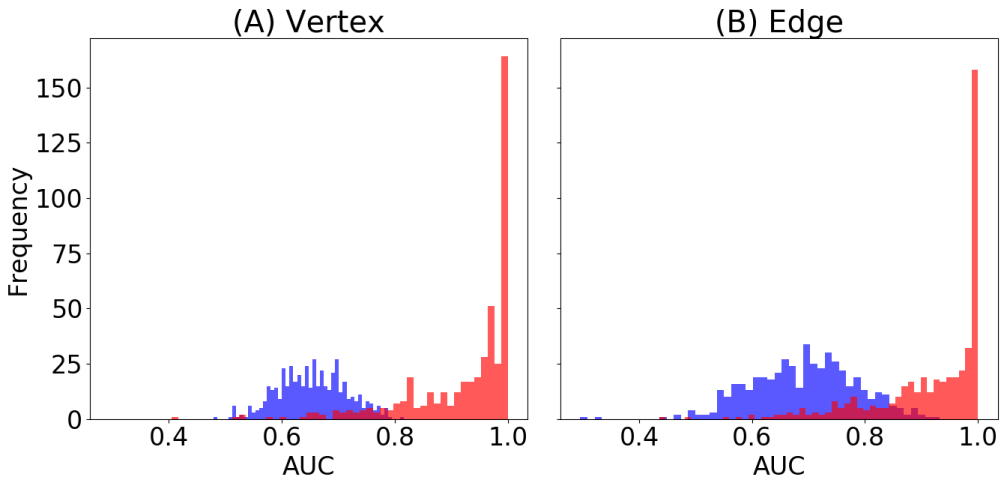

Figure 6.2.: Histograms of area under the ROC curve (AUC) evaluated on 487 proteins in the DynDom dataset. (A) Histograms of AUC calculated from the inter- and intra-vertices in the line graphs derived from the protein graph(blue histogram) and from the coarse-grained graph (red histogram). (B) Histograms of AUC calculated from the inter- and intra-edges in the line graphs derived from the protein graph(blue histogram) and from the coarse-grained graph (red histogram).

Too small clusters, on the other hand, tend to have smaller inconsistency errors with the cost of the insignificance of the mean variance between two clusters.

6.1.2. Line Graph Transformation

Here, I deliver my insights and motivations concerning the line graph transformation. First, I explain why I need to modify the construction of the line graph as to make it suitable to my study of rigid domains in protein estimation. Second, I present the motivation beneath the formula of feature functions defined on vertices in the line graph ($\Psi^{(1)}$ in Equation 4.1.7). Afterward, I discuss my ideas behind the formula of feature functions defined on edges in the line graph ($\Psi^{(2)}$ in Equations 4.1.8, 4.1.9, 4.1.10) as well as their limitations.

Modified Line Graph Construction

In the old line graph construction [79], any pair of incident edges in the original will become an edge in a line graph. In case of the two ended vertices of those pair incident edges also form an edge, the mean-variance of these two vertices is used twice in the quality function, one for a vertex feature and one for an edge feature. To avoid such duplication, I modified the line graph construction by eliminate edges if their two end vertices are linked. Additionally, such edges pruning in the line graph produces a sparser graph which saves computational resources in the calculation of the generalized Viterbi algorithm.

Feature functions on vertices in the line graph

Given a protein with known rigid domains, the left panel of Figure 6.1.2 shows that the mean-variance values of inter-vertices in the line graph constructed from the coarse-grained graph tend to be bigger than ones of the intra-vertices. This observation fits very well with the rigidity definition. However, an optimal threshold dividing mean-variance of inter- and intra-vertices is protein-dependent and there is no such universal threshold. Yet I noticed that if I consider the mean-variance of inter-vertices as outliers, I could identify almost of them via outliers detection. Thus, in the feature functions on vertices, I reward the quality function when either the mean-variance of a vertex is an outlier (probably an inter-vertex) and the predicted label of that vertex is -1, or the mean-variance of a vertex is non-outlier (probably an intra-vertex) and its predicted label is $+1$.

Feature functions on edges in the line graph

With the similar observation, the right panel in Figure 6.1.2 shows that the mean-variance of inter- and intra-edges in the line graph seems to follow two distinguish but overlapped distributions. Still, it is problematic to calculate these two distributions because there are not

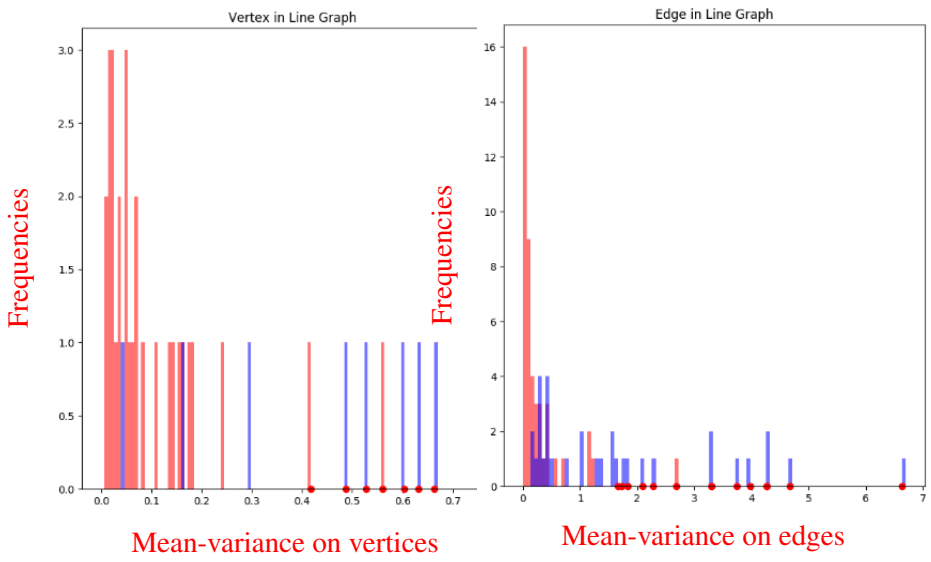

Figure 6.3.: The histograms of mean-variance values calculated from vertices and edges in a line graph. (Left) The left panel shows the frequencies distribution of the mean variance calculated on intra-vertices (red bars) and inter-vertices (blue bars) in the line graph. The big red dots the x-axis indicate the outliers according to the outliers detection. (Right) Similarly, the right panel illustrates the frequencies distribution of mean-variance for the intra- and inter-edges in the line graph. The big red dots are also the outliers according to the outliers detection.

enough samples for the maximum likelihood estimator such as expectation maximization (EM). However, if I applied the outliers detection trick as mentioned above, I could obtain a decent amount of inter-edges based on their mean-variance. As shown in the right panel of Figure 6.1.2, outliers indicated via big red dots in the x-axis cover a lot of mean-variance calculated from the inter-edges.

From such observations, I designed a feature function for an edge in the line graph in a way that it could be in favor of vertices labels according to the mean-variance of the edge and its two corresponding vertices. The design of the feature function on edges $\Psi^{(2)}$ is based on the inferences described on Figure 6.4. From Figures 6.4.A to 6.4.D, it is trivial that a preferable labeling could be obtained directly from the mean-variance of edges and their two vertices. For instance, in Figure 6.4.A, a high mean-variance of two ended vertices along with a low and a high mean-variance between a common vertex and two ended vertices infer a positive and negative labels of vertices in the line graph, respectively. The labeling inferences of other cases such as (B), (C) and (D) follow the similar reasoning. However, there are two cases (6.4.E&F) where the method could not unambiguously infer the labeling.

Those unambiguous cases happen when signals from two pairs of vertices show they belong to one domain but the signal from other pair shows that they do not. Nevertheless, when I examined those two cases, there were substantial differences.

In the case of $\gamma_e = -1$ and $\gamma_{v_1} = \gamma_{v_2} = +1$ (Figure 6.4.E), it implies that two end vertices probably belong to different domains and the common vertex locates on the hinge region. To decide which domain this common vertex belongs to, two mean-variance of v_1 and v_2 were compared, as shown in the Equation 4.1.9. On the other hand, Figure 6.4.F shows another contradictory case where I set the value to zero, thus they could not interrupt to the labeling inference of the generalized Viterbi algorithm.

6.1.3. Running Time

The algorithm running time depends on several folds. First, the protein size plays a big role in terms of time for the construction of protein graph, coarse-grained graph as well as the calculation of mean-variance. Second, the densities of a protein graph and a coarse-grained graph also affect a lot to the running time. The running time of the generalized Viterbi algorithm especially depends on how dense a line graph is. In a lot of cases, the exact most probable labeling was impossible to be obtained and thus heuristics have been applied. Third, the algorithm running time also heavily depends upon the rigidity of conformational changes. In the small structural transition, the signals derived from the mean-variance fail to decisively distinguish between inter- and intra-vertices and edges. Consequently, it required the algorithm to run on the graph multiple times.

Figure 6.5 summarizes the relationship between protein sizes and their running time. In general, the running time for proteins smaller than 800 amino acids increases slightly in a linear manner. Yet it seems to quadratically grow for the large proteins. It is noticeable that there are a few outlier proteins whose running time are significantly longer than of ones with similar size. In those cases, the mean-variance between inter- and intra-vertices/edges in a line graph were totally overlap and thus resulting in bad quality function. This caused the algorithm run multiple iterations and only broke when it reached the limitation.

6.1.4. Merging Algorithm

The post-process such as merging is necessary in the study of rigid domains in proteins estimation because it avoids the advents of very small and fragmented domains. One may reasonably ask whether users could skip the labeling step (the inference of the Viterbi algorithm on a line graph constructed from a coarse-grained graph) and apply the merging process directly on clusters from the coarse graining. This simplified version of my algorithm has shown good results on proteins with large-scale movements and less flexibility in their rigid domains, but failed on general cases.

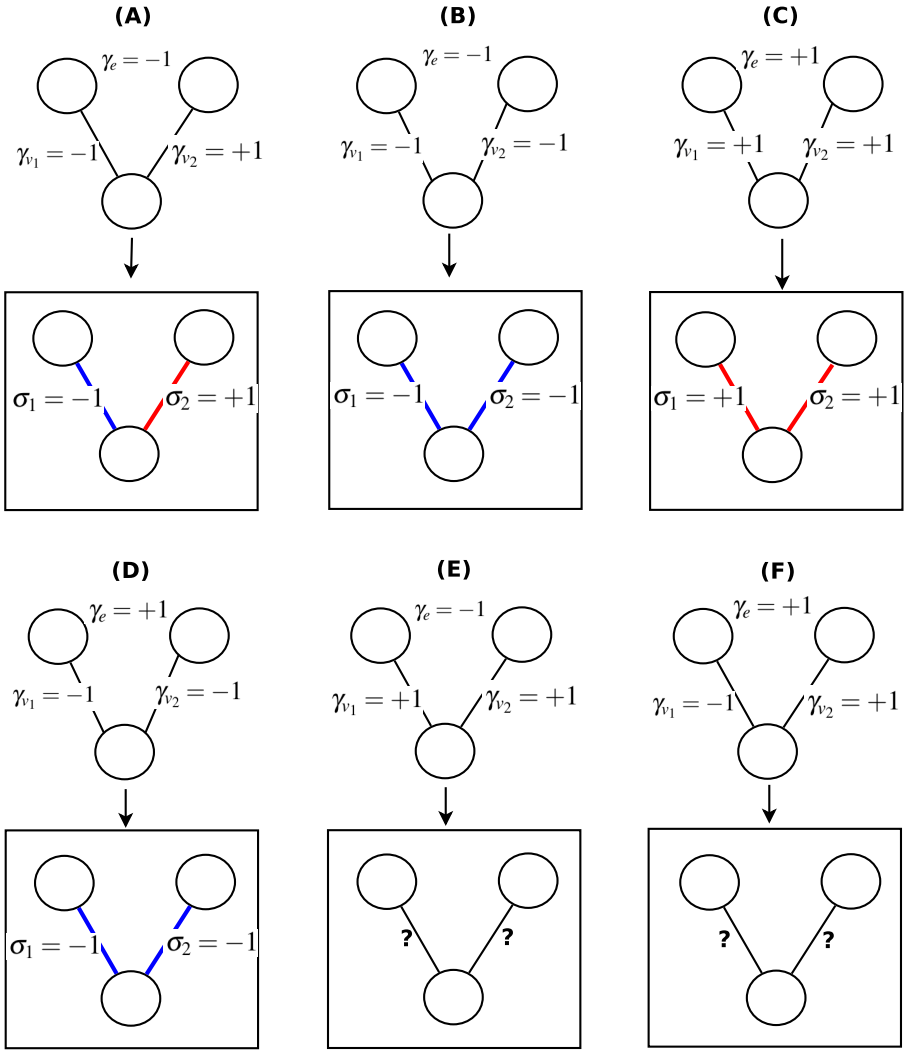

Figure 6.4.: An edge $e = (v_1, v_2)$ in the line graph represented by a pair of edges in the original graph. From (**A**) to (**D**): the labels of two vertices in the line graph (edges in the original graph) are unambiguously determined through mean-variances of two ended nodes of an edge and two vertices in the line graph. (**E**) & (**F**): an ambiguity of labeling two vertices in the line graph occurs when there are two signals indicating that three nodes should belong to one domain, but the other signal suggests the otherwise.

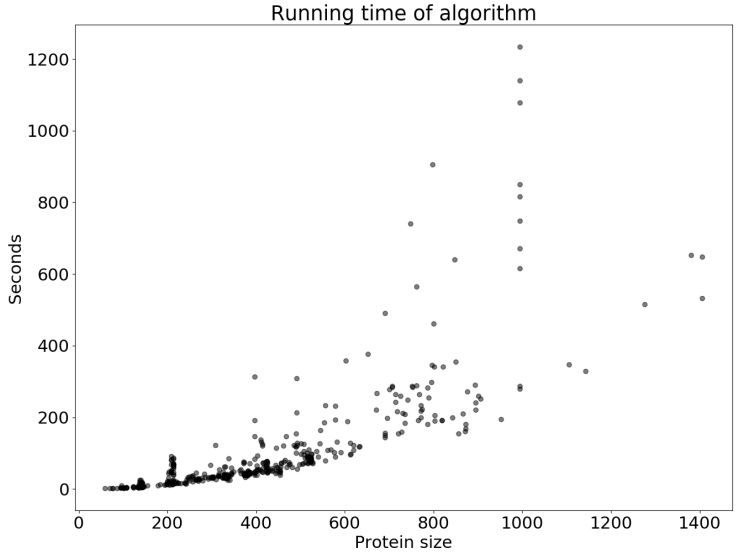

Figure 6.5.: Protein size versus running time (measured in seconds) evaluated for 487 proteins selected from the DynDom database.

6.2. Answers for Research Questions Concerning the Task of Novel Sequence-based Feature Engineering

The results from Section 5.2 show that the advent of the information theory-based feature $\mathfrak{f}_{\mathbb{JSD}}$ clearly boosts the performance of the RF classifier in identifying the DNA-binding sites in proteins when it is combined with existing features such as $\mathfrak{f}_{\mathbb{PSSM}}$, $\mathfrak{f}_{\mathbb{OBV}}$ and $\mathfrak{f}_{\mathbb{SS}}$. This section is based on the Discussion section in our paper [25] (see Appendix A.2).

In spite of both MSAs derived feature, my new feature and PSSMs are substantially different because they carry distinct kinds of evolutionary information. The PSSM feature which is a 20D vector computes a statistic of how likely an amino acid occurs at a certain position, meanwhile the JSD-based feature takes into account the divergences of a distribution of pairs of amino acids to a null distribution constructed from known non-binding sites. Even though the JSD-based feature is only a single scalar, it significantly improve the performance when concatenating with other existing high dimensional features.

In the RF classifier setting, the number m of randomly selected features (see the description of RF classifier in Section 3.2.1 and Algorithm 3.2) depends on the problem and should be treated as a tuning parameter [97]. This parameter influences the generalization error of RF in two ways: strength of an individual tree and the correlations among trees. When m is

too big (close to the number of total features P), the strengths of trained trees are high, yet their correlations are also big which negatively affects to the RF performance. On the other hand, if m is too small, the forest contains all weak but very small correlated trees. According to Breiman [64], the number of used features in the classification problem is about the square root of the total features ($m = \lfloor P \rfloor$) [64, 97]. The author of RF [64] also suggested $m = \lfloor \log_2 P + 1 \rfloor$ [64]. In the study of DNA-binding sites in proteins prediction, I noticed that the first option ($\lfloor P \rfloor$) offered a better result, meanwhile other values of m around $\lfloor P \rfloor$ gave similar results. Thus, I set $\lfloor P \rfloor$ as the default parameter.

The number of trees in a forest is also an important parameters. As mentioned in [97], the RF classifier stabilized at about 200 trees. In my study, this classifier began to stabilize around 100 trees which was also the value I used for my study of DNA-binding sites prediction.

In DNA-protein complexes, only a small portion of proteins are directly responsible for the interaction with DNA. Thus, the class imbalance is arisen due to the fact that the number of DNA-binding sites in proteins is significantly less than the number of nonbinding sites. To deal with such imbalance issue, one could use data-driven techniques, algorithm-driven techniques, or the combination of these two. The data-driven techniques employ data sampling methods to limit the affect of the data imbalance, meanwhile the algorithm-driven techniques use weights, cost schema as well as adapting underlying classifiers and their outputs as to avoid the bias toward the majority class [98].

In my study of DNA-binding sites in proteins prediction, I used data sampling methods to reduce the imbalance effect between positive and negative samples. The proportion of negative samples in each bootstrap sampling depends on how much sensitivity and specificity of the designed algorithm. I noticed that I obtained the best result (according to MCC) when the number of negative samples in the training set is from twice to three times than the number of positive ones. Additionally, the number of RF classifiers in my method could be any arbitrary odd number. Nevertheless, it reached the stability with eleven RF classifiers in the method.

7. Conclusion

In this section, I conclude the thesis. For this, I provide a short summary and give an outlook on my potential future work.

7.1. Summary

In the course of this thesis, I present two methods which I developed aiming to solve the identifications of rigid domains as well as DNA-binding sites in proteins in a computational fashion.

In the study of rigid domains in proteins detection, I introduced a new algorithm to characterize structural changes in proteins. The new graph-based algorithm comprises several stages such as constructing a protein graph from multiple conformations, reducing graph complexity via the coarse graining, inferring the binary labeling of edges through a line graph transformation along with the generalized Viterbi algorithm. The crucial feature of this new method is that the number of rigid domains is learnt automatically. Yet users could relax the rigidity definition, thus be able to attain the desirable number of rigid domains. Overall, my segmentations and other methods such as DynDom [14], Spectrus [18], Habeck&Nguyen [19] have a strong agreement on various medium to large scale structural transitions.

In the second part of the thesis which studies DNA-binding sites in proteins, I propose a new sequence-based feature for such binding sites detection. My new feature applies Jensen–Shannon divergence to quantify the differences between the observed amino acids distribution of sites and the null distribution constructed from the amino acids distribution of non-binding sites. The results from several large benchmarks offer a strong evidence that the combination of existing features with my new sequence-based feature significantly improve the predictions of the RF classifier.

7.2. Outlook

In regard to the graph-based models of rigid domains in proteins detection, there are many aspects which are well worth being taken into consideration. First, the quality function consisting of feature functions on vertices and edges (Equations $\Psi^{(1)}$, $\Psi^{(2)}$) could be designed

to integrate relevant information. As mentioned in the Discussion section, users could integrate their prior segmentations to help the coarse graining to reduce the error of inconsistency, thus resulting in the improvement of the rigid domains detection. Additionally, other useful information such as amino acids charge and secondary structure could be embedded in the algorithm either by modifying the graph construction (as described in the Discussion section) or via the designs of feature functions on vertices and edges. Second, instead of returning binary values, the feature functions could be designed to return continuous values which are beneficial for the analysis.

Concerning the study of identifying DNA-binding sites in proteins, it has been shown that a carefully designed doubly stochastic matrix could improve the predictions of DNA-binding sites in proteins [25]. Moreover, I was able to improve the effect of doubly stochastic matrix by harvesting biochemical signals which distinguishes binding and non-binding residues. Additionally, the results from Tables 5.8, 5.9, 5.10 and 5.11 have shown that incorporating my new sequence-based feature with one existing feature such as PSSM significantly improves the performance. Yet this positive effect is gradually faded with the advent of other features such as secondary structure (SS) and orthogonal binary vector (OBV). Thus, there is a room to improve the performance by eliminating the redundancy in the feature space through a careful feature selection.

Bibliography

[1] C. N. Pace, J. M. Scholtz, and G. R. Grimsley, "Forces stabilizing proteins," *FEBS Letters*, vol. 588, no. 14, pp. 2177–2184, May 2014. [Online]. Available: https://doi.org/10.1016/j.febslet.2014.05.006

[2] "Chapter 9 - antibodies," in *Immunology for Pharmacy*, D. K. Flaherty, Ed. Saint Louis: Mosby, 2012, pp. 70 – 78. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780323069472100094

[3] A. Blanco and G. Blanco, "Chapter 8 - enzymes," in *Medical Biochemistry*, A. Blanco and G. Blanco, Eds. Academic Press, 2017, pp. 153 – 175. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780128035504000082

[4] S. Martínez Cuesta, S. A. Rahman, N. Furnham, and J. M. Thornton, "The classification and evolution of enzyme function," *Biophysical journal*, vol. 109, no. 6, pp. 1082–1086, Sep 2015, 25986631[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/25986631

[5] R. G. Smith, *Overview of Human Growth Hormone*. Totowa, NJ: Humana Press, 2000, pp. 1–13. [Online]. Available: https://doi.org/10.1007/978-1-59259-015-5_1

[6] M. M. Qaid and M. M. Abdelrahman, "Role of insulin and other related hormones in energy metabolism—a review," *Cogent Food & Agriculture*, vol. 2, no. 1, p. 1267691, 2016. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/23311932.2016.1267691

[7] M. D. Shoulders and R. T. Raines, "Collagen structure and stability," *Annual review of biochemistry*, vol. 78, pp. 929–958, 2009, 19344236[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/19344236

[8] G. C. Yeo, B. Aghaei-Ghareh-Bolagh, E. P. Brackenreg, M. A. Hiob, P. Lee, and A. S. Weiss, "Fabricated elastin," *Advanced healthcare materials*, vol. 4, no. 16, pp. 2530–2556, Nov 2015, 25771993[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/25771993

[9] M. F. PERUTZ, M. G. ROSSMANN, A. F. CULLIS, H. MUIRHEAD, G. WILL, and A. C. T. NORTH, "Structure of hæmoglobin: A three-dimensional fourier synthesis at 5.5-å. resolution, obtained by x-ray analysis," *Nature*, vol. 185, no. 4711, pp. 416–422, 1960. [Online]. Available: https://doi.org/10.1038/185416a0

[10] Z. Dong, K. Wang, T. K. Linh Dang, M. Gültas, M. Welter, T. Wierschin, M. Stanke, and S. Waack, "Crf-based models of protein surfaces improve protein-protein interaction site predictions," *BMC Bioinformatics*, vol. 15, no. 1, p. 277, Aug 2014. [Online]. Available: https://doi.org/10.1186/1471-2105-15-277

[11] K. Henzler-Wildman and D. Kern, "Dynamic personalities of proteins," *Nature*, vol. 450, pp. 964–972, 2007.

[12] M. Gerstein, A. M. Lesk, and C. Chothia, "Structural mechanisms for domain movements in proteins," *Biochemistry*, vol. 33, no. 22, pp. 6739–6749, 1994.

[13] T. K. L. Dang, T. Nguyen, M. Habeck, M. Gültas, and S. Waack, "A graph-based algorithm for detecting rigid domains in protein structures," *BMC Bioinformatics*, vol. 22, no. 1, Feb. 2021. [Online]. Available: https://doi.org/10.1186/s12859-021-03966-3

[14] S. Hayward and H. Berendsen, "Systematic analysis of domain motions in proteins from conformational change: new results on citrate synthase and t4 lysozyme: New results on citrate synthase and t4 lysozyme," *Proteins*, vol. 30, no. 2, pp. 144 – 154, 1998, journal EB 1 V342 ROTEIN-STRUCT FUNCT GENET.

[15] W. Wriggers and K. Schulten, "Protein domain movements: detection of rigid domains and visualization of hinges in comparisons of atomic coordinates," *Proteins: Structure, Function, and Bioinformatics*, vol. 29, no. 1, pp. 1–14, 1997. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/%28SICI%291097-0134%28199709%2929%3A1%3C1%3A%3AAID-PROT1%3E3.0.CO%3B2-J

[16] W. Kabsch, "A solution for the best rotation to relate two sets of vectors," *Acta Crystallographica Section A*, vol. 32, pp. 922–923, Sep. 1976.

[17] A. Abyzov, R. Bjornson, M. Felipe, and M. Gerstein, "Rigidfinder: a fast and sensitive method to detect rigid blocks in large macromolecular complexes," *PROTEINS: Structure, Function, and Bioinformatics*, vol. 78, no. 2, pp. 309–324, 2010.

[18] L. Ponzoni, G. Polles, V. Carnevale, and C. Micheletti, "Spectrus: A dimensionality reduction approach for identifying dynamical domains in protein complexes from limited structural datasets," *Structure*, vol. 23, no. 8, pp. 1516 – 1525, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0969212615002270

[19] T. Nguyen and M. Habeck, "A probabilistic model for detecting rigid domains in protein structures," *Bioinformatics*, vol. 32, no. 17, pp. i710–i717, 08 2016. [Online]. Available: https://doi.org/10.1093/bioinformatics/btw442

[20] J. Vogelgesang and M. Scharkow, "Bayesian statistics," pp. 1–9, Nov. 2017. [Online]. Available: https://doi.org/10.1002/9781118901731.iecrm0013

[21] B. E. Trumbo and E. A. Suess, "Gibbs sampling," in *Encyclopedia of Social Network*

Analysis and Mining. Springer New York, 2018, pp. 942–957. [Online]. Available: https://doi.org/10.1007/978-1-4939-7131-2_146

[22] U. Emekli, D. Schneidman-Duhovny, H. J. Wolfson, R. Nussinov, and T. Haliloglu, "Hingeprot: automated prediction of hinges in protein structures," *Proteins: Structure, Function, and Bioinformatics*, vol. 70, no. 4, pp. 1219–1227, 2008.

[23] K. Hinsen, "Analysis of domain motions by approximate normal mode calculations," *Proteins: Structure, Function, and Bioinformatics*, vol. 33, no. 3, pp. 417–429, 1998.

[24] S. C. Flores and M. B. Gerstein, "Flexoracle: predicting flexible hinges by identification of stable domains," *BMC bioinformatics*, vol. 8, no. 1, p. 215, 2007.

[25] T. K. L. Dang, C. Meckbach, R. Tacke, S. Waack, and M. Gültas, "A novel sequence-based feature for the identification of dna-binding sites in proteins using jensen-shannon divergence," *Entropy*, vol. 18, p. 379, 2016.

[26] X. Ma, J. Guo, H. Liu, J. Xie, and X. Sun, "Sequence-based prediction of dna-binding residues in proteins with conservation and correlation information," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 9, no. 6, pp. 1766–1775, 2012.

[27] J. Wu, H. Liu, X. Duan, Y. Ding, H. Wu, Y. Bai, and X. Sun, "Prediction of DNA-binding residues in proteins from amino acid sequences using a random forest model with a hybrid feature," *Bioinformatics*, vol. 25, no. 1, pp. 30–35, 11 2008. [Online]. Available: https://doi.org/10.1093/bioinformatics/btn583

[28] J. Si, Z. Zhang, B. Lin, M. Schroeder, and B. Huang, "Metadbsite: a meta approach to improve protein dna-binding sites prediction," *BMC systems biology*, vol. 5 Suppl 1, no. Suppl 1, pp. S7–S7, Jun 2011, 21689482[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/21689482

[29] L. Wang and S. J. Brown, "Bindn: a web-based tool for efficient prediction of dna and rna binding sites in amino acid sequences," *Nucleic acids research*, vol. 34, no. Web Server issue, pp. W243–W248, Jul 2006, 16845003[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/16845003

[30] Y. Ofran, V. Mysore, and B. Rost, "Prediction of DNA-binding residues from sequence," *Bioinformatics*, vol. 23, no. 13, pp. i347–i353, 07 2007. [Online]. Available: https://doi.org/10.1093/bioinformatics/btm174

[31] C. Yan, M. Terribilini, F. Wu, R. L. Jernigan, D. Dobbs, and V. Honavar, "Predicting dna-binding sites of proteins from amino acid sequence," *BMC Bioinformatics*, vol. 7, no. 1, p. 262, May 2006. [Online]. Available: https://doi.org/10.1186/1471-2105-7-262

[32] S. Hwang, Z. Gou, and I. B. Kuznetsov, "DP-Bind: a web server for

sequence-based prediction of DNA-binding residues in DNA-binding proteins," *Bioinformatics*, vol. 23, no. 5, pp. 634–636, 01 2007. [Online]. Available: https://doi.org/10.1093/bioinformatics/btl672

[33] Z. Miao and E. Westhof, "Prediction of nucleic acid binding probability in proteins: a neighboring residue network based score," *Nucleic Acids Research*, vol. 43, no. 11, pp. 5340–5351, 05 2015. [Online]. Available: https://doi.org/10.1093/nar/gkv446

[34] L. Wang, C. Huang, M. Q. Yang, and J. Y. Yang, "Bindn+ for accurate prediction of dna and rna-binding residues from protein sequence features," *BMC Systems Biology*, vol. 4, no. 1, p. S3, May 2010. [Online]. Available: https://doi.org/10.1186/1752-0509-4-S1-S3

[35] S. Ahmad, M. M. Gromiha, and A. Sarai, "Analysis and prediction of dna-binding proteins and their binding residues based on composition, sequence and structural information," *Bioinformatics*, vol. 20, no. 4, p. 477–486, Mar. 2004. [Online]. Available: https://doi.org/10.1093/bioinformatics/btg432

[36] S. Ahmad and A. Sarai, "Pssm-based prediction of dna binding sites in proteins," *BMC Bioinformatics*, vol. 6, no. 1, p. 33, Feb 2005. [Online]. Available: https://doi.org/10.1186/1471-2105-6-33

[37] K.-C. Wong, Y. Li, C. Peng, A. M. Moses, and Z. Zhang, "Computational learning on specificity-determining residue-nucleotide interactions," *Nucleic Acids Research*, vol. 43, no. 21, pp. 10 180–10 189, 11 2015. [Online]. Available: https://doi.org/10.1093/nar/gkv1134

[38] L. Wang, M. Q. Yang, and J. Y. Yang, "Prediction of dna-binding residues from protein sequence information using random forests," *BMC genomics*, vol. 10 Suppl 1, no. Suppl 1, pp. S1–S1, Jul 2009, 19594868[pmid]. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/19594868

[39] V. A. Traag, P. Van Dooren, and Y. Nesterov, "Narrow scope for resolution-limit-free community detection," *Physical Review E*, vol. 84, no. 1, pp. +016 114, Jul. 2011.

[40] D. N. Wilson, A. G. Hinnebusch, T. E. Dever, and N. Sonenberg, *Initiation of Protein Synthesis*. John Wiley & Sons, Ltd, 2006, ch. 7, pp. 219–322. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/3527603433.ch7

[41] F. Crick, *Chapter 8: The genetic code*, 1990, p. 89–101.

[42] M. Remmert, A. Biegert, A. Hauser, and J. Söding, "Hhblits: lightning-fast iterative protein sequence searching by hmm-hmm alignment," *Nature Methods*, vol. 9, no. 2, pp. 173–175, Feb 2012. [Online]. Available: https://doi.org/10.1038/nmeth.1818

[43] S. Altschul, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," *Nucleic Acids Research*, vol. 25, no. 17, pp. 3389–3402, Sep. 1997.

[Online]. Available: https://doi.org/10.1093/nar/25.17.3389

[44] D. Frishman and P. Argos, "Seventy-five percent accuracy in protein secondary structure prediction," *Proteins: Structure, Function, and Genetics*, vol. 27, no. 3, pp. 329–335, Mar. 1997. [Online]. Available: https://doi.org/10.1002/(sici)1097-0134(199703)27:3<329::aid-prot1>3.0.co;2-8

[45] S. A. Coulocheri, D. G. Pigis, K. A. Papavassiliou, and A. G. Papavassiliou, "Hydrogen bonds in protein–dna complexes: Where geometry meets plasticity," *Biochimie*, vol. 89, no. 11, pp. 1291 – 1303, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0300908407001964

[46] J. Shen, J. Zhang, X. Luo, W. Zhu, K. Yu, K. Chen, Y. Li, and H. Jiang, "Predicting protein-protein interactions based only on sequences information," *Proceedings of the National Academy of Sciences*, vol. 104, no. 11, pp. 4337–4341, Mar. 2007. [Online]. Available: https://doi.org/10.1073/pnas.0607879104

[47] Y. Xu, S. Furao, J. Zhao, and O. Hasegawa, "To obtain orthogonal feature extraction using training data selection," 01 2009, pp. 1819–1822.

[48] D. Cai and X. He, "Orthogonal locality preserving indexing," in *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 3–10. [Online]. Available: https://doi.org/10.1145/1076034.1076039

[49] Huan Liu and Lei Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491–502, 2005.

[50] M. Steinegger, M. Meier, M. Mirdita, H. Vöhringer, S. J. Haunsberger, and J. Söding, "HH-suite3 for fast remote homology detection and deep protein annotation," *BMC Bioinformatics*, vol. 20, no. 1, Sep. 2019. [Online]. Available: https://doi.org/10.1186/s12859-019-3019-7

[51] "Protein data bank contents guide: Atomic coordinate entry format description version 3.30," *wwPDB*, 2012. [Online]. Available: ftp://ftp.wwpdb.org/pub/pdb/doc/format_descriptions/Format_v33_Letter.pdf

[52] M. Pocock, T. Down, and T. Hubbard, "Biojava: Open source components for bioinformatics," *SIGBIO Newsl.*, vol. 20, no. 2, p. 10–12, Aug. 2000. [Online]. Available: https://doi.org/10.1145/360262.360266

[53] P. J. A. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski, and M. J. L. de Hoon, "Biopython: freely available python tools for computational molecular biology and bioinformatics," *Bioinformatics*, vol. 25, no. 11, pp. 1422–1423, Mar. 2009. [Online]. Available:

https://doi.org/10.1093/bioinformatics/btp163

[54] G. Qi, R. Lee, and S. Hayward, "A comprehensive and non-redundant database of protein domain movements," *Bioinformatics*, vol. 21, no. 12, pp. 2832–2838, Mar. 2005. [Online]. Available: https://doi.org/10.1093/bioinformatics/bti420

[55] Y. Zhang and J. Skolnick, "Scoring function for automated assessment of protein structure template quality," *Proteins: Structure, Function, and Bioinformatics*, vol. 57, no. 4, pp. 702–710, 2004. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.20264

[56] Y. Zhang, "Tm-align: a protein structure alignment algorithm based on the TM-score," *Nucleic Acids Research*, vol. 33, no. 7, pp. 2302–2309, Apr. 2005. [Online]. Available: https://doi.org/10.1093/nar/gki524

[57] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 7 1948. [Online]. Available: https://ieeexplore.ieee.org/document/6773024/

[58] R. Eggeling, T. Roos, P. Myllymäki, and I. Grosse, "Inferring intra-motif dependencies of dna binding sites from chip-seq data," *BMC Bioinformatics*, vol. 16, no. 1, p. 375, 2015. [Online]. Available: https://doi.org/10.1186/s12859-015-0797-4

[59] M. Gültas, G. Düzgün, S. Herzog, S. J. Jäger, C. Meckbach, E. Wingender, and S. Waack, "Quantum coupled mutation finder: predicting functionally or structurally important sites in proteins using quantum jensen-shannon divergence and cuda programming," *BMC Bioinformatics*, vol. 15, no. 1, p. 96, 2014. [Online]. Available: https://doi.org/10.1186/1471-2105-15-96

[60] J. D. Fischer, C. E. Mayer, and J. Söding, "Prediction of protein functional residues from sequence by probability density estimation," *Bioinformatics*, vol. 24, no. 5, pp. 613–620, 01 2008. [Online]. Available: https://doi.org/10.1093/bioinformatics/btm626

[61] J. A. Capra and M. Singh, "Predicting functionally important residues from sequence conservation," *Bioinformatics*, vol. 23, no. 15, pp. 1875–1882, 05 2007. [Online]. Available: https://doi.org/10.1093/bioinformatics/btm270

[62] M. A. Ré and R. K. Azad, "Generalization of entropy based divergence measures for symbolic sequence analysis," *PLOS ONE*, vol. 9, no. 4, pp. 1–11, 04 2014. [Online]. Available: https://doi.org/10.1371/journal.pone.0093532

[63] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. USA: Wiley-Interscience, 2006.

[64] L. Breiman, *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. [Online]. Available: https://doi.org/10.1023/a:1010933404324

[65] A.-L. Boulesteix, G. Tutz, and K. Strimmer, "A CART-based approach to discover emerging patterns in microarray data," *Bioinformatics*, vol. 19, no. 18, pp. 2465–2472, 12 2003. [Online]. Available: https://doi.org/10.1093/bioinformatics/btg361

[66] A. Papana and H. Ishwaran, "CART variance stabilization and regularization for high-throughput genomic data," *Bioinformatics*, vol. 22, no. 18, pp. 2254–2261, 07 2006. [Online]. Available: https://doi.org/10.1093/bioinformatics/btl384

[67] G. Stiglic, S. Kocbek, I. Pernek, and P. Kokol, "Comprehensive decision tree models in bioinformatics," *PLOS ONE*, vol. 7, no. 3, pp. 1–13, 03 2012. [Online]. Available: https://doi.org/10.1371/journal.pone.0033812

[68] X. Chen, M. Wang, and H. Zhang, "The use of classification trees for bioinformatics," *Wiley interdisciplinary reviews. Data mining and knowledge discovery*, vol. 1, no. 1, pp. 55–63, Jan 2011, 22523608[pmid]. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/22523608

[69] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks, 1984.

[70] L. Breiman, "Technical note: Some properties of splitting criteria," *Machine Learning*, vol. 24, no. 1, pp. 41–47, 1996. [Online]. Available: https://doi.org/10.1023/a:1018094028462

[71] S. España-Boquera, M. J. Castro-Bleda, F. Zamora-Martínez, and J. Gorbe-Moya, "Efficient viterbi algorithms for lexical tree based models," in *Advances in Nonlinear Speech Processing*, M. Chetouani, A. Hussain, B. Gas, M. Milgram, and J.-L. Zarader, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 179–187.

[72] J. Felsenstein, "Maximum Likelihood and Minimum-Steps Methods for Estimating Evolutionary Trees from Data on Discrete Characters," *Systematic Biology*, vol. 22, no. 3, pp. 240–249, 09 1973. [Online]. Available: https://doi.org/10.1093/sysbio/22.3.240

[73] A. T. Ihler, J. W. Fischer III, and A. S. Willsky, "Loopy belief propagation: Convergence and effects of message errors," *J. Mach. Learn. Res.*, vol. 6, p. 905–936, Dec. 2005.

[74] T. Wierschin, K. Wang, M. Welter, S. Waack, and M. Stanke, "Combining features in a graphical model to predict protein binding sites," *Proteins: Structure, Function, and Bioinformatics*, vol. 83, no. 5, pp. 844–852, 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.24775

[75] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, oct 2008. [Online]. Available: https://doi.org/10.1088%2F1742-5468%2F2008%2F10%2Fp10008

[76] J. Reichardt and S. Bornholdt, "Partitioning and modularity of graphs with arbitrary degree distribution," *Phys. Rev. E*, vol. 76, p. 015102, Jul 2007. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.76.015102

[77] B. Iglewicz and D. C. Hoaglin, *How to detect and handle outliers*. Asq Press, 1993, vol. 16.

[78] V. A. Traag, "Faster unfolding of communities: Speeding up the Louvain algorithm," *Physical Review E*, vol. 92, no. 3, pp. +032 801, sep 2015.

[79] T. S. Evans and R. Lambiotte, "Line graphs of weighted networks for overlapping communities," *The European Physical Journal B*, vol. 77, no. 2, pp. 265–272, 2010. [Online]. Available: https://doi.org/10.1140/epjb/e2010-00261-8

[80] I. Grosse, P. Bernaola-Galván, P. Carpena, R. Román-Roldán, J. Oliver, and H. E. Stanley, "Analysis of symbolic sequences using the jensen-shannon divergence," *Physical Review E*, vol. 65, no. 4, Mar. 2002. [Online]. Available: https://doi.org/10.1103/physreve.65.041905

[81] A. R. Panchenko, "Prediction of functional sites by analysis of sequence and structure conservation," *Protein Science*, vol. 13, no. 4, pp. 884–892, Apr. 2004. [Online]. Available: https://doi.org/10.1110/ps.03465504

[82] J.-O. Janda, M. Busch, F. Kück, M. Porfenenko, and R. Merkl, "CLIPS-1d: analysis of multiple sequence alignments to deduce for residue-positions a role in catalysis, ligand-binding, or protein structure," *BMC Bioinformatics*, vol. 13, no. 1, p. 55, 2012. [Online]. Available: https://doi.org/10.1186/1471-2105-13-55

[83] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, pp. 10–18, Nov. 2009. [Online]. Available: https://doi.org/10.1145/1656274.1656278

[84] C. Müller, G. Schlauderer, J. Reinstein, and G. Schulz, "Adenylate kinase motions during catalysis: an energetic counterweight balancing substrate binding," *Structure*, vol. 4, no. 2, pp. 147 – 156, 1996. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0969212696000184

[85] B. Glader, "Chapter 72 - other hereditary red blood cell disorders," in *Emery and Rimoin's Principles and Practice of Medical Genetics*, D. Rimoin, R. Pyeritz, and B. Korf, Eds. Oxford: Academic Press, 2013, pp. 1 – 25. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123838346000768

[86] R. A. Lee, M. Razaz, and S. Hayward, "The DynDom database of protein domain motions," *Bioinformatics*, vol. 19, no. 10, pp. 1290–1291, 07 2003. [Online]. Available: https://doi.org/10.1093/bioinformatics/btg137

[87] J. Xu and Y. Zhang, "How significant is a protein structure similarity with TM-score

= 0.5?" *Bioinformatics*, vol. 26, no. 7, pp. 889–895, Feb. 2010. [Online]. Available: https://doi.org/10.1093/bioinformatics/btq066

[88] K. Lim, R. J. Read, C. C. H. Chen, A. Tempczyk, M. Wei, D. Ye, C. Wu, D. Dunaway-Mariano, and O. Herzberg, "Swiveling domain mechanism in pyruvate phosphate dikinase,," *Biochemistry*, vol. 46, no. 51, pp. 14 845–14 853, 2007, pMID: 18052212. [Online]. Available: https://doi.org/10.1021/bi701848w

[89] K. Theis, P. Gong, and C. T. Martin, "Topological and conformational analysis of the initiation and elongation complex of t7 rna polymerase suggests a new twist," *Biochemistry*, vol. 43, no. 40, pp. 12 709–12 715, 2004, pMID: 15461442. [Online]. Available: https://doi.org/10.1021/bi0486987

[90] D. C. Boisvert, J. Wang, Z. Otwinowski, A. L. Norwich, and P. B. Sigler, "The 2.4 å crystal structure of the bacterial chaperonin groel complexed with atpgs," *Nature Structural Biology*, vol. 3, no. 2, pp. 170–177, 1996. [Online]. Available: https://doi.org/10.1038/nsb0296-170

[91] A. Karmen, F. Wroblewski, and J. S. Ladue, "Transaminase activity in human blood," *The Journal of clinical investigation*, vol. 34, no. 1, pp. 126–131, Jan 1955, 13221663[pmid]. [Online]. Available: https://www.ncbi.nlm.nih.gov/pubmed/13221663

[92] N. E and W. HJ, "Diphosphopyridinproteid ackohol, acetaldehyd," *Biochem*, vol. 293, no. 351, 1937.

[93] H. THEORELL and J. S. McKINLEY McKEE, "Mechanism of action of liver alcohol dehydrogenase," *Nature*, vol. 192, no. 4797, pp. 47–50, 1961. [Online]. Available: https://doi.org/10.1038/192047a0

[94] T. Li, Q.-Z. Li, S. Liu, G.-L. Fan, Y.-C. Zuo, and Y. Peng, "PreDNA: accurate prediction of DNA-binding sites in proteins by integrating sequence and geometric structure information," *Bioinformatics*, vol. 29, no. 6, pp. 678–685, 01 2013. [Online]. Available: https://doi.org/10.1093/bioinformatics/btt029

[95] A. Krall, J. Brunn, S. Kankanala, and M. H. Peters, "A simple contact mapping algorithm for identifying potential peptide mimetics in protein–protein interaction partners," *Proteins: Structure, Function, and Bioinformatics*, vol. 82, no. 9, pp. 2253–2262, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.24592

[96] S. K. Nair and S. K. Burley, "X-ray structures of myc-max and mad-max recognizing dna: Molecular bases of regulation by proto-oncogenic transcription factors," *Cell*, vol. 112, no. 2, pp. 193–205, Jan 2003. [Online]. Available: https://doi.org/10.1016/S0092-8674(02)01284-9

[97] T. Hastie, J. Friedman, R. Tisbshirani, and undefined, *Random forests*. Springer,

2017, p. 587–602.

[98] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, Mar. 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0192-5

Acronyms

ADK Adenylate Kinase.

AhD Alcohol dehydrogenase.

AST Aspartate aminotransferase.

CART Classification and Regression Tree.

CPM Constant Potts Model.

CRF Conditional Random Field.

JSD Jensen–Shannon divergence.

KL Kullback–Leibler.

MCC Matthews correlation coefficient.

MSA multiple sequence alignment.

OBV orthogonal binary vectors.

PDB Protein Data Bank.

PPDK Pyruvate phosphate dikinase.

PSSM position-specific scoring matrix.

RF Random Forest.

RMSD root-mean-square deviation.

SS secondary structure.

SVM support vector machine.

Glossary

antibodies

A protein produced by cells of the immune system that binds to antigens. 1, 79

antigens

anything that causes your body to make antibodies; antigens could be foreign blood cells, a toxin, bacteria, or the cells of a transplanted organ.. 1, 79

coarse graining

a procedure to reduce the protein graph to coarse-grained graph.. 4, 40, 41, 55–57, 61, 83

coarse-grained graph

a reduced version of the original graph where its vertices are groups of vertices in the original graph.. 29, 31, 79

common vertex

a vertex shared by two incident edges.. 30, 32, 60, 81

communities

groups of vertices in the protein graph. 29, 31

DNA

Deoxyribonucleic acid (DNA): The genetic material of organisms, usually double-stranded; a class of nucleic acids identified by the presence of deoxyribose, a sugar, and the four nucleobases.. 1

domain

A region of a protein responsible for a particular function, as recognized experimentally and by the occurrence of similar segments in other proteins sharing that function, e.g., a DNA binding domain.. 1

enzymes

proteins (or rarely, RNA) that catalyze a chemical reaction.. 1

error of inconsistency

the metric quantifies the heterogeneity of communities in the coarse-grained graph weighted by their sizes.. 41, 56, 80

feature function

a function defined on a vertex or an edge in the site graph.. 31, 32

inconsistency error

another name of error of inconsistency.. 55, 56

inter edge

an edge in a line graph is call inter edge if its corresponding two ended vertices in the original graph belong to different domains.. 30

inter vertex

a vertex in a line graph is called inter vertex if the two corresponding vertices in the original graph belong to different domains.. 30, 31

intra edge

an edge in a line graph is call intra edge if its corresponding two ended vertices in the original graph belong to an identical domain.. 30

intra vertex

a vertex in a line graph is called intra vertex if the two corresponding vertices in the original graph belong to an identical domain.. 30, 31

line graph

a line graph is a graph resulted in the transformation from the original graph where edges in the original graph become vertices in the line graph.. 4, 27, 30, 31

mean variance

a means to measure the fluctuation between two groups of vertices in the protein graph.. 29, 30

null background distribution

a background distribution calculated according to amino acid counts at non-binding residue positions in MSAs.. 5, 35

outliers detection

an algorithm detects outliers from a set of real-value numbers.. 30

peptide bond

a covalent bond formed by joining the carboxyl group of one amino acid to the amino group of another, with the removal of a molecule of water.. 1

profile

A set of input variables used in making predictions.. 3

protein graph

a graph constructed by multiple conformations of a protein where each vertex is an amino acid.. 4, 27–29, 31, 79

quality function

the unnormalized probability of a given labeling of a site graph in CRF.. 31, 32

reduced graph

another name of the coarse-grained graph.. 4, 29

signal

the separability between the mean variance values of inter- and intra-vertices or edges in the line graph.. 55

sliding window

an odd length window whose center is the target amino acid.. 35

transcription factors proteins

Proteins that have DNA-binding domains and thus have a specific or general affinity for single- or double-stranded DNA.. 1

two ended vertices

the other vertices (without a common vertex)of two incident edges.. 30, 32, 60, 80

List of Figures

2.1. Chemical structure of an amino acid with *R* group 7

2.2. Peptide bond between two amino acids 8

3.1. An undirected graph. 21

4.1. Sliding windows with radius $n = 3$ in three scenarios. The target residues are colored in red and their corresponding sliding windows in blue. (A) When the target residue is close to the beginning of the sequence, the left wing of the window is partially missed. (B) A full sliding window of a target residue. (C) A sliding window of an ended residue, resulting in missing its right wing. 36

5.1. Graph-based segmentation of ADK into rigid domains. (A) A protein graph constructed from open and closed ADK conformations. (B) a reduced/coarse-grained graph obtained by coarse graining the protein graph. (C) A line graph of the reduced graph. (D) The line graph with binary vertex labels (black: -1, white: +1) obtained via the generalized Viterbi algorithm. (E) The injective relation between edges of the reduced graph and vertices of the line graph allows us to also label the edges of the reduced graph. Edges with negative labels are removed, resulting to three disconnected subgraphs. (F) A segmented protein graph derived from disconnected subgraphs in the reduced graph. (G) ADK graph with domain annotation from literature encoded by colors. 40

5.2. The histogram of the error and overlap evaluated on 487 proteins in the DynDom data set. 45

5.3. Protein graph of human importin subunit beta-1 protein. (**A**) Segmentation suggested by DynDom: three rigid domains colored by dark green, red and blue. (**B**) My segmentation: two rigid domains colored by light green and blue. 46

5.4. Analysis of several proteins undergoing conformational changes on a variety of scales. Large-scale conformational changes: pyruvate phosphate dikinase, T7 RNA polymerase, GroEL. Medium-scale conformational changes: Aspartate aminotransferase, Alcohol dehydrogenase. For each protein, the segmentation found by different methods and in the literature are shown. Same color means same domain. 47

5.5. DNA-binding sites in proto-oncobenic transcription factor MYC-MAX protein complex (PDB-Entry 1NKP). Green spheres denote positions of the DNA-binding sites in both proteins which are detected by RF classifier either using the existing features ($\mathfrak{f}_{\mathbb{PSSM}}$, $\mathfrak{f}_{\mathbb{OBV}}$, and $\mathfrak{f}_{\mathbb{SS}}$) alone or combining my new features with these existing features together. Purple spheres show the localization of additional binding sites which were only found by RF classifier using my new features with existing features. Moreover, there are further three binding sites in MYC protein and one binding site in MAX protein, shown with yellow spheres, that could not be identified by the classifier. 53

6.1. Histogram of inconsistency error from graph construction type I and II and their varying cutoff values respectively. 57

6.2. Histograms of area under the ROC curve (AUC) evaluated on 487 proteins in the DynDom dataset. (A) Histograms of AUC calculated from the inter- and intra-vertices in the line graphs derived from the protein graph(blue histogram) and from the coarse-grained graph (red histogram). (B) Histograms of AUC calculated from the inter- and intra-edges in the line graphs derived from the protein graph(blue histogram) and from the coarse-grained graph (red histogram). 58

6.3. The histograms of mean-variance values calculated from vertices and edges in a line graph. (Left) The left panel shows the frequencies distribution of the mean variance calculated on intra-vertices (red bars) and inter-vertices (blue bars) in the line graph. The big red dots the x-axis indicate the outliers according to the outliers detection. (Right) Similarly, the right panel illustrates the frequencies distribution of mean-variance for the intra- and inter-edges in the line graph. The big red dots are also the outliers according to the outliers detection. 60

6.4. An edge $e = (v_1, v_2)$ in the line graph represented by a pair of edges in the original graph. From (**A**) to (**D**): the labels of two vertices in the line graph (edges in the original graph) are unambiguously determined through mean-variances of two ended nodes of an edge and two vertices in the line graph. (**E**) & (**F**): an ambiguity of labeling two vertices in the line graph occurs when there are two signals indicating that three nodes should belong to one domain, but the other signal suggests the otherwise. 62

6.5. Protein size versus running time (measured in seconds) evaluated for 487 proteins selected from the DynDom database. 63

List of Algorithms

3.1. Pseudo code of CART algorithm. 16
3.2. Pseudo code of Random Forest classifiers. 17
3.3. Pseudo code of the generalized Viterbi algorithm. 20
4.1. Pseudo code of my graph-based method to calculate rigid domains in proteins. 33

List of Tables

3.1. Values of feature function $\Psi^{(1)}$ defined on vertices. 21
3.2. Values of feature function $\Psi^{(2)}$ defined on edges. 22
3.3. Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{a\}$. . . 22
3.4. Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{a,b\}$. . 22
3.5. Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{b,c\}$. . 23
3.6. Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{b,c,d\}$. 23
3.7. Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{c,e\}$. . 23
3.8. Values of the *logquality* function defined on the boundary set $\mathcal{B} = \{f\}$. . . 24

5.1. An example of a protein with two segmentations. 42
5.2. Inter-intra relationships over all pair of amino acids in the 1^{st} segmentation. 42
5.3. Inter-intra relationships over all pair of amino acids in the 2^{nd} segmentation. 43
5.4. The table contains disagreements between Tables 5.2 and 5.3. 43
5.5. The overlap table between two segmentations. 44
5.6. The performance of my graph-based method with varying edge cutoff values assessed on DynDom data set. 46
5.7. Proteins in different scale conformational changes involved in the assessment. 48
5.8. Prediction performance of RF classifier on different features using a cut-off of 3.5 Å. The prediction system was evaluated by five-fold cross validation. 50
5.9. Prediction performance of Random Forest (RF) classifier on different features using a cut-off of 5.0 Å. The prediction system was evaluated by five-fold cross validation. 50
5.10. Prediction performance of RF classifier on RBscore data set using different distance cut-offs. 51
5.11. Prediction performance of RF classifier on PreDNA dataset using different distance cut-offs. 52
5.12. Prediction performance of RF classifier on different features using a cut-off of 3.5 Å for MYC-MAX protein complex (Protein Data Bank (PDB)-Entry 1NKP). 54

A. Appendix

A.1. The Calculation of Rigid Domains in Proteins

Dang *et al. BMC Bioinformatics* (2021) 22:66
https://doi.org/10.1186/s12859-021-03966-3

BMC Bioinformatics

METHODOLOGY ARTICLE Open Access

A graph-based algorithm for detecting rigid domains in protein structures

Truong Khanh Linh Dang[1*], Thach Nguyen[2], Michael Habeck[2,3,4], Mehmet Gültas[5,6] and Stephan Waack[1]

*Correspondence: linh.dang@informatik.uni-goettingen.de
[1] Institute of Computer Science, University of Göttingen, Goldschmidtstr 7, 37077 Göttingen, Germany
Full list of author information is available at the end of the article

Abstract

Background: Conformational transitions are implicated in the biological function of many proteins. Structural changes in proteins can be described approximately as the relative movement of rigid domains against each other. Despite previous efforts, there is a need to develop new domain segmentation algorithms that are capable of analysing the entire structure database efficiently and do not require the choice of protein-dependent tuning parameters such as the number of rigid domains.

Results: We develop a graph-based method for detecting rigid domains in proteins. Structural information from multiple conformational states is represented by a graph whose nodes correspond to amino acids. Graph clustering algorithms allow us to reduce the graph and run the Viterbi algorithm on the associated line graph to obtain a segmentation of the input structures into rigid domains. In contrast to many alternative methods, our approach does not require knowledge about the number of rigid domains. Moreover, we identified default values for the algorithmic parameters that are suitable for a large number of conformational ensembles. We test our algorithm on examples from the DynDom database and illustrate our method on various challenging systems whose structural transitions have been studied extensively.

Conclusions: The results strongly suggest that our graph-based algorithm forms a novel framework to characterize structural transitions in proteins via detecting their rigid domains. The web server is available at http://azifi.tz.agrar.uni-goettingen.de/webservice/.

Keywords: Protein structural transition, Graph algorithms, Generalized Viterbi algorithm

Background

Proteins are molecular machines that are involved in a large variety of biological processes. Protein function is often driven by large-scale structural transitions [1]. Experimental methods for biomolecular structure determination such as X-ray crystallography, NMR and cryo-electron microscopy have been used to determine thousands of atomic structures of proteins in different conformational states. A powerful approach to understand structural transitions in proteins is to decompose structures of different states into rigid domains and classify protein movements by hinge and shear motions of these structural domains [2].

© The Author(s) 2021. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

Given the large number of available protein structures, we need computational methods that identify structurally conserved domains in a set of alternative structures in an automated fashion with minimal user intervention. For example, one could use the software to study molecular dynamics trajectories at the level of rigid domains to gain an understanding of large-scale movements, or identify important active sites located at the interface between rigid domains.

A number of computational methods for detecting rigid domains in protein structures have been developed. Dyndom [3] identifies rigid domains by clustering a set of rotation vectors. Hingefind [4] focuses on the detection of hinge residues, which are detected via differences in bending angles. RigidFinder [5] finds rigid domains via a dynamic programming algorithm that optimizes the rigidity of structural segments extracted from two conformational states. These methods are limited to two input structures and require the selection of a cutoff parameter [5], which can impact the results quite strongly. Spectrus [6] applies spectral clustering to distance fluctuations and supports multiple input structures. However, the number of clusters relies on a quality score, which sometimes gives ambiguous results. Probabilistic approaches [7, 8] segment protein structures into rigid domains as part of a generative probabilistic model. The model parameters, including the segmentation, are inferred with expectation maximization or Gibbs sampling. However, choosing the initial parameters as well as the number of rigid segments is still a critical issue, because both algorithms explore parameter space only locally, and can therefore require many restarts from different initial conditions.

A more ambitious goal is to predict rigid domains from a single structure by, for example, molecular dynamic simulation or an elastic network model that can both be used to generate a set of alternative conformational states. HingeProt [9] and Domain Finder [10] use an elastic network model to predict hinge residues by analyzing the correlation between selected pairs of eigenvectors of the correlation matrix. However, in general it is unclear which modes contribute most strongly to the movement, in particular if a conformational change involves multiple modes. FlexOracle [11] finds hinge positions by identifying split points with minimal energetic impact.

Despite the rich literature on methods for rigid-domain detection in protein structures, all of the existing methods require the initial number of rigid domains in their calculation. Thus, there is still a need for algorithms that are robust, reliable, able to handle high-throughput data and yet do not require extensive parameter tunning. Here, we introduce a graph-based method that infers a binary labeling that encodes if pairs of amino acids belong to identical or different rigid domains. Our algorithm proceeds in two stages: first, we construct a protein graph based on spatial proximity, which we cluster using the Louvain algorithm to obtain a coarse-grained graph of reduced size. Second, edges in the reduced graph are labeled by applying a line graph transformation along with the general Viterbi algorithm. We benchmark our algorithm on 487 entries of the DynDom database and find a high agreement with the reference segmentation. In addition, we also present a detailed analysis of various proteins that show a large variety of conformational transitions and compare our results to other methods.

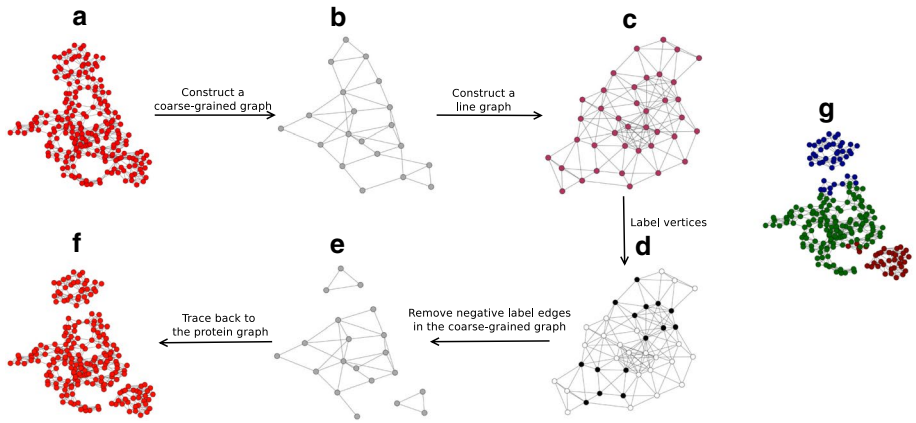

Fig. 1 Graph-based segmentation of ADK into rigid domains. **a** Protein graph constructed from open and closed conformations. **b** Reduced graph obtained by coarse-graining the protein graph. **c** A line graph of the reduced graph. **d** A line graph with binary vertex labels (black: − 1, white: + 1) obtained with the generalized Viterbi algorithm. **e** The injective relation between edges of the reduced graph and vertices of the line graph allows us to also label the edges of the reduced graph. Edges having negative labels are removed resulting in three disconnected subgraphs. **f** A segmented protein graph derived from disconnected subgraphs in the reduced graph. **g** ADK graph with domain annotation from literature encoded by colors

Results

To validate our algorithm, we first segment conformations of Adelynate Kinase (ADK). We then perform a benchmark on 487 proteins from the DynDom database. Finally, we compare our method with other domain segmentation algorithms on a number of test cases ranging from medium to large scale conformational changes.

Rigid segmentation of Adenylate Kinase

We first run our algorithm for rigid domain segmentation on Adenylate Kinase (ADK) for which multiple experimental structures showing different conformations are available [12]. ADK catalyzes the interconversion of adenine nucleotides and is composed of three rigid domains. By closing the NMP-binding domain and the LID domain onto the CORE domain, ADK binds ATP and AMP which are converted to two ADP molecules. The PDB codes of ADK open and closed conformations are 4ake and 1ake (both chain A) respectively. ADK is composed of 214 amino acids which constitute the vertices of the initial protein graph. To build the protein graph from both states, we used $\delta = 7.5$ Å as cutoff.

Figure 1 illustrates the workflow of our algorithm and intermediate results for ADK using default values for the algorithmic parameters. Figure 1a shows ADK's protein graph in which each vertex is an amino acid; the construction of edges linking spatially close amino acids is described in Methods. Amino acids are grouped by running the Louvain domain detection algorithm [13] and merged into vertices of a coarse-grained graph. In the case of ADK, the protein graph comprising 214 vertices is transformed to a coarse-grained graph composed of 20 vertices (Fig. 1b). In the next step, we construct the line graph of the coarse-grained graph (Fig. 1c). We then run the generalized Viterbi algorithm [14] on a scoring function defined on the line graph. This results in a binary labeling of the line graph (Fig. 1d) or, equivalently, a labeling of the coarse-grained

graph. Based on this labeling our method splits the coarse-grained graph into three disconnected subgraphs (Fig. 1e). Finally, we map the unconnected subgraphs back to the protein graph to obtain a segmentation of ADK into three rigid domains (Fig. 1f). Our segmentation agrees strongly with the domain boundaries defined in the literature [15], which we color-coded in Fig. 1g for visual comparison. Our segmentation deviates from the literature annotation only in the hinge regions. This discrepancy is due to the ambiguous membership of amino acids in the hinge region which tend to be merged with amino acids from different domains in the coarse-graining step.

Unlike DynDom, our method also works with multiple conformational states. To study this feature, we ran our algorithm again but on 100 ADK conformations generated by morphing between the open and closed state [16]. The algorithm produces a similar segmentation.

An advantage of our method is that it allows users to integrate prior knowledge to improve the segmentation. For example, for the default parameter setting, our method incorrectly assigned fifteen amino acids of the NMP-binding and LID domain to the core domain. Yet with some prior knowledge about the rigid domains, we can improve the rigid-domain segmentation. Suppose we are given ADK's segmentation calculated from Spectrus [6] with $K = 4$ (number of rigid domains). We can integrate this prior knowledge into our model as follows. The weights of edges in the protein graph whose vertices belong to different domains according to the prior labeling are reduced by a factor $\alpha < 1$. Here, we choose $\alpha = 0.75$. This setting helps the coarse-graining process to reduce the error of inconsistency (mentioned in the Discussion) and thus improve the performance. We then ran our graph-based method on the new coarse-grained graph and found that only five amino acids of the LID domain were wrongly assigned to the core domain. Thus even imperfect prior knowledge can significantly improve the result.

Rigid segmentation benchmark

We benchmarked our method on the DynDom database [17] reduced to those pairs of proteins whose overall RMSD exceeds 5 Å. Moreover, we removed domains that span less than ten amino acids. To evaluate our method, we use the segmentation error and overlap defined by [8]. The overlap counts the number of matches between two segmentations after solving a low-dimensional linear assignment problem that maximizes the agreement between the two labelings. The error assesses how often two segmentations disagree on whether a pair of amino acids belongs to the same domain. Although both metrics differ in the details, they are highly anti-correlated.

Figure 2 shows histograms of the error and overlap between our and DynDom's segmentation evaluated on 487 proteins based on an edge cutoff value of 7.5 Å. The median error is 0.038 and the median overlap 0.972. The error and overlap histograms are highly skewed to small and large values, respectively. For approximately 30% of the examples, our method reaches a near perfect agreement with the annotation provided by DynDom (overlap ≥ 0.99). In only a few cases our method fails to produce a reasonable segmentation due to errors in the coarse-graining step and/or an indistinguishable signal derived from the mean variance. Despite of the disagreements between our method and DynDom, our segmentation sometimes seems to be more reasonable. We investigate the open and closed states of human importin subunit beta-1 (PDB code 3lww, chains A and

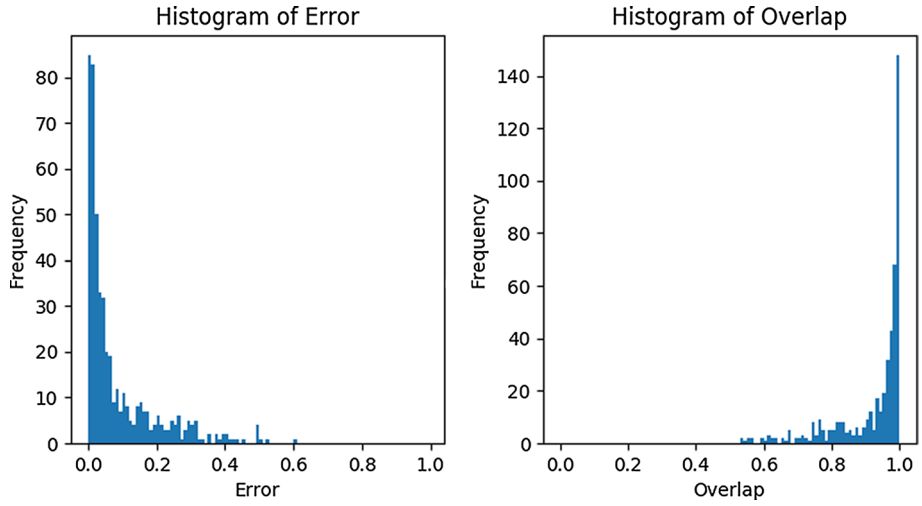

Fig. 2 Histogram of the error and the overlap evaluated on 487 proteins in the DynDom database

C) as an example. According to Dyndom, this protein has three rigid domains (Fig. 3a) whose RMSDs are 6.8, 4.3, and 2.1 Å, respectively. We note that the first domain found by DynDom (dark green) is small, fragmented and shows a large RMSD. A large portion of the second domain (dark red) is interspersed with the third domain (dark blue). Our segmentation suggests two separate domains whose RMSDs are 2.2 and 1.0 Å (Fig. 3b), which are much smaller than the RMSDs produced by DynDom's segmentation.

To study the impact of the edge cutoff used in the definition of the protein graph, we ran experiments with varying cutoff values. Table 1 reports the mean and median of the overlap and error obtained with different edge cutoff values. The overlap seems to be largely unaffected by the specific choice of the cutoff, whereas the error drops slightly with larger cutoffs. Two possible explanations come to our mind. First, a larger cutoff results in protein graphs with more connections between amino acid vertices. Denser graphs seem to be more suitable to coarse graining with the Louvain method (see Additional file 1: Figure S1 and the Discussion for a demonstration of this claim). Second, also the coarse-grained graph will be denser with larger cutoff values, which seems to improve the scoring of the line graph. However, because denser graphs result in larger line graphs, we need to restrict the cutoff to smaller values to tame the computational costs of the Viterbi algorithm.

Analysis of various structural transitions

We ran our method on various proteins studied in [8] showing different types and scales of conformational changes. Table 2 provides the protein name, size and PDB code; Fig. 4 shows a summary of the segmentation analysis. First, we study and compare the performance of our algorithm (graph-based method) to other methods by analyzing protein complexes that undergo large-scale conformational changes.

Pyruvate phosphate dikinase (PPDK) is a large biomolecular complex that catalyzes the reversible conversion of PEP, AMP, and P_i to pyruvate and ATP [18]. We apply our graph-based method to two PPDK structures and compare the segmentation to the

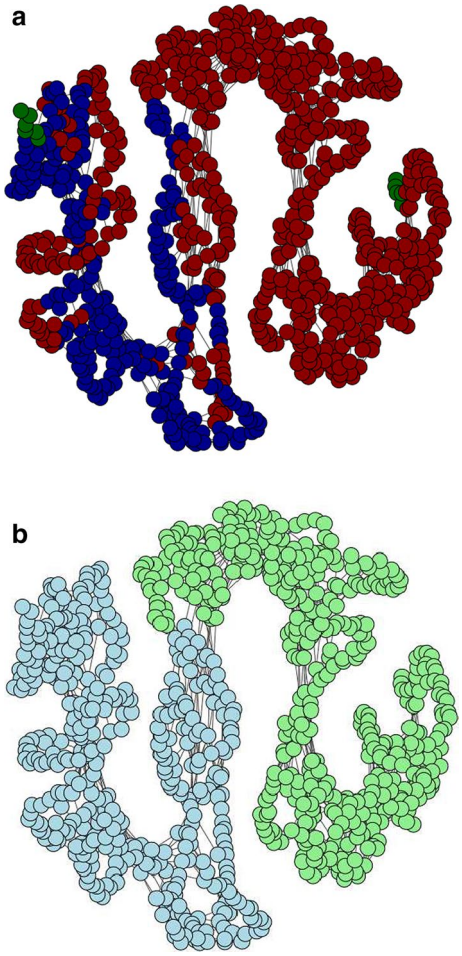

Fig. 3 Protein graph of human importin subunit beta-1 protein. **a** Segmentation suggested by DynDom: three rigid domains colored in dark green, red and blue. **b** Segmentation estimated by our method: two rigid domains colored in light green and blue

Table 1 Performance of the graph-based algorithm for different edge cutoffs evaluated on the DynDom benchmark

Cutoff (Å)	Metric			
	Median overlap	Mean overlap	Median error	Mean error
7.5	0.972	0.924	0.038	0.086
10.5	0.977	0.924	0.034	0.083
13.5	0.972	0.926	0.033	0.081

annotation found in the literature [18] and by other methods such as Spectrus, DynDom as well as Nguyen&Habeck2016 [8]. Our segmentation agrees strongly with the segmentation provided by DynDom, but fails to detect the additional domain reported in the literature and by [8]. Typically, our method produces a smaller number of domains than reported in the literature, because we only take changes in a few structural snapshots into account and no additional experimental information. For $K = 3$, Spectrus agrees

Table 2 Proteins in different scale conformational changes involved in the assessment

Protein	PDB code	Chain ID	Size
PPDK	1kc7	A	872
	2r82	A	
T7 RNA polymerase	1qln	A	842
	1msw	D	
GroEL	1aon	A	524
	1aon	H	
Aspatate aminotransferase	9aat	A	401
	1ama	A	
Alcohol dehydrogenase	1adg	A	374
	2ohx	A	

strongly with the segmentation found by our graph-based approach except for the first domain, which is significantly larger according to Spectrus.

T7 RNA polymerase is involved in the initiation and elongation of RNA transcription. Our segmentation is highly consistent with the results from DynDom, [8] and the anotation from the literature [19]. Spectrus fails to identify the refolding loop inserted in the N-terminal domain.

The chaperonin GroEL [20] provides a shielded environment to assist protein folding and prevent aggregation. For this example, all methods provide very similar segmentation results.

We also benchmark our method on proteins undergoing medium-scale structural transitions. Aspartate aminotransferase (AST) is an enzyme involved in amino acid metabolism that catalyzes the reversible transfer of an α-amino group between aspartate and glutamate [21]. For this example, we find a high agreement between our method and other segmentations. Another example is the enzyme Alcohol dehydrogenase (AhD) that decomposes alcohol into aldehyde. Our graph-based segmentation agrees strongly with the result from DynDom. Spectrus achieves its maximum score for $K = 3$ domains, but introduces an additional domain compared to the other methods. For $K = 2$, the score is lower, but Spectrus' segmentation is more consistent to DynDom and our result.

Discussion

Our results demonstrate that segmentation of protein conformations into rigid domains can be achieved with a graph-based algorithm that solves the rigid segmentation problem with an edge-labeling strategy. Let us discuss the key features of the algorithm and the impact of algorithmic parameters. To measure the efficiency of the graph construction and coarse graining, we use a metric that we call inconsistency error. The inconsistency error quantifies the heterogeneity of clusters weighted by their size. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph composed of $N = |\mathcal{V}|$ vertices $v_i \in \mathcal{V}$ with labels σ_i and $\mathcal{C} = \{\mathcal{C}_k\}$ a partition of the vertices into clusters $\mathcal{C}_k \subset \mathcal{V}$ obtained by coarse graining. We define the inconsistency error of the coarse graining procedure as $\mathrm{error}(\mathcal{C}|\mathcal{G}) = 2\sum_{\mathcal{C}_k \in \mathrm{C}} \frac{|\mathcal{C}_k|}{\mathrm{N}} \frac{\sum_{i<j \in \mathcal{C}_k} |\sigma_i \neq \sigma_j|}{|\mathcal{C}_k|\,(|\mathcal{C}_k|-1)}$ which is the average number of labeling mismatches within each cluster weighted by cluster size.

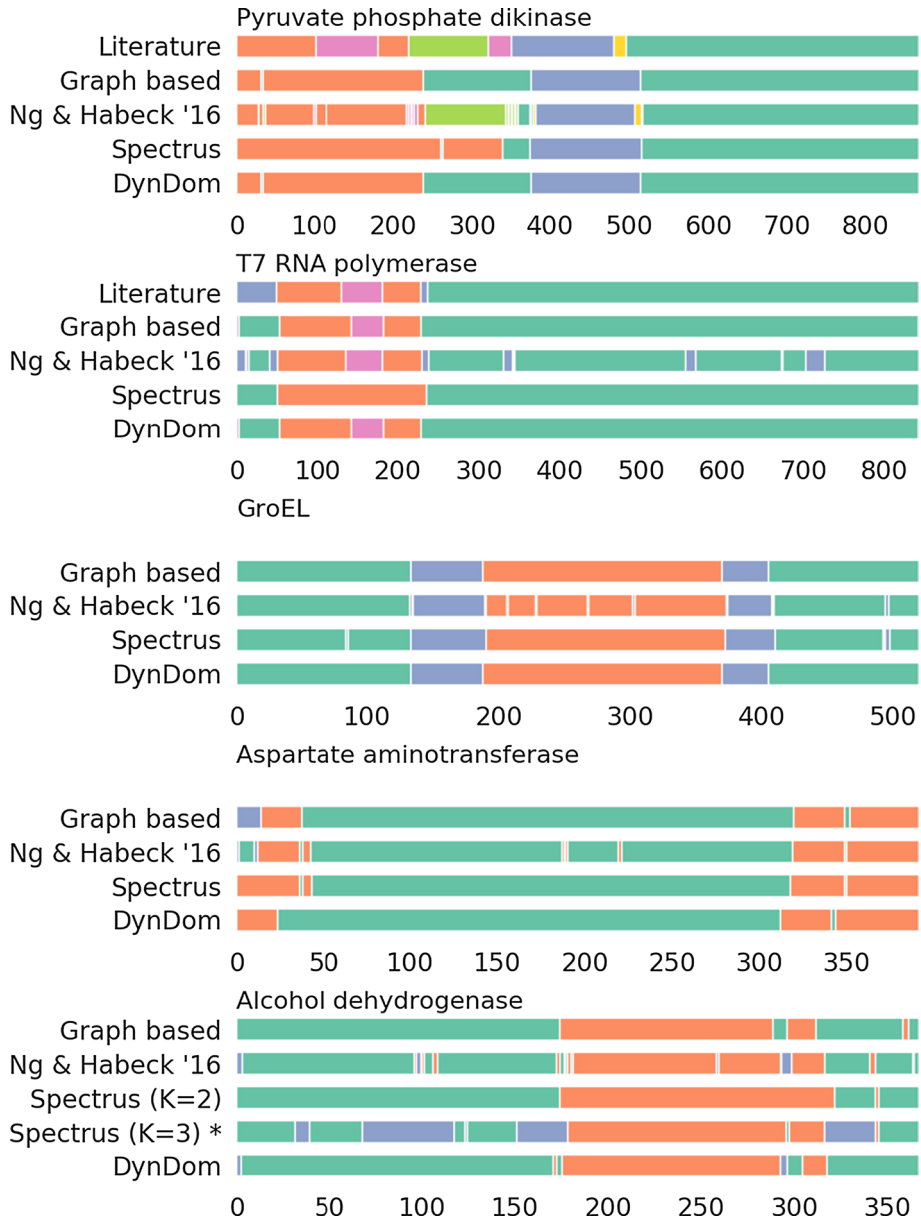

Fig. 4 Analysis of several proteins undergoing conformational changes on a variety of scales. Large-scale conformational changes: pyruvate phosphate dikinase, T7 RNA polymerase, GroEL. Medium-scale conformational changes: Aspartate aminotransferase, Alcohol dehydrogenase. For each protein, the segmentation found by different methods and in the literature are shown. Same color means same domain

We first study different ways to construct a protein graph from multiple conformations. There are many reasonable options for constructing a protein graph. For example, one possibility is to create an edge if the distance between two vertices is smaller than a cutoff in at least one conformation, and to assign as a weight the number of such conformations. Another possibility (detailed in Methods) is to create an edge if its distance is smaller than the cutoff in *all* conformations, and to weight the edge by the reciprocal exponentiated variance computed over all conformations (such that low-variance edges

have a weight close to one and large-variance edges are assigned small weights). Additional file 1: Figure S1 demonstrates that the second graph construction rule consistently outperforms the first rule based on the inconsistency error. We therefore used the second rule in our benchmark calculations. In addition, we tested different values of the edge cutoff distance and noticed a minor, but not significant improvement of the inconsistency error for larger cutoff values.

We also studied various options for the coarse-graining step. In all tests, we used the Louvain algorithm for fitting Potts models [13] for coarse graining. The resolution parameter was adjusted so as to produce about 20 clusters of medium size. Too large clusters risk to merge amino acids from hinge regions and thus the inconsistency error is expected to increase. Too small clusters will tend to show a smaller inconsistency error at the cost of lowering the significance of the mean variance between two clusters. Large graphs will pose a computational challenge in the Viterbi step, because the number of vertices of the line graph grows quadratically with the number of vertices in the original graph. By using our coarse-graining strategy, we save computational resources and enhance the signal as shown in Additional file 1 (see second section and Additional file 1: Figure S2).

Moreover, we ran our algorithm on Lysozyme [22], an enzyme contributing to the innate immune system, to investigate if this graph-based algorithm could produce a reasonable segmentation given several actual conformations. In this study, we use 100 conformations of Lysozyme whose PDB codes can be found in the Supplementary Information. To account for minor differences in the protein sequences, we align all proteins with Clustal Omega Alignment (https://www.ebi.ac.uk/Tools/msa/clustalo/). Our segmentation on Lysozyme completely agrees with Spectrus [6] and Nguyen&Habeck2016 [8] where all methods suggest two domains whose *RMSD*s are 1.6 and 4.9 Å, respectively.

Our method is also applicable to study rigid domains in membrane proteins. For instance, the chemokine receptor CCR5 [23] located on the surface of white blood cells plays an important role in the immune system. Here, we consider various conformational states of CCR5 (PDB codes: 6aky_A, 4mbs_A, 6akx_A, 5uiw_A). The sequences of these four conformational states were aligned with Clustal Omega [24, 25]. Our segmentation finds a small (51 amino acids 223–253) and a big (286 amino acids 1–222 & 254–337) rigid domain whose *RMSD*s are 0.6 and 1.6 Å, respectively. This segmentation is stable against variations in the rigidity threshold and does not require the execution the merging procedure. When we reduced the threshold to define the protein graph to 4.5 Å, we obtained two different domains: a small domain (amino acids 193–246) and a large domain (amino acids 1–192 and 247–337) whose RMSDs deteriorated to 2.7 and 3.0 Å, respectively.

To avoid duplication of features involving vertices and edges, we modify the construction of the line graph by discarding an edge if its two end vertices are connected as well. That way, features extracted from edges add new information. Finally, we use a merging routine with heuristic criteria to merge two domains. One may ask if we could skip the labeling step (Viterbi algorithm) and apply the merging routine directly to the clusters found by coarse graining. This simplified version of our algorithm achieves good results on proteins showing a large-scale movement, but fails on more subtle cases. Overall,

post-processing via the merging procedure compensates for segmentation errors involving small fragments.

The running time of our algorithm depends on the size of the protein, the density of the protein graph, and the rigidity of the conformational change. Additional file 1: Figure S3 shows the relationship between protein size and the running time of our graph-based segmentation algorithm. We note that the running time for proteins smaller than 800 amino acids grows slowly in a linear fashion. For the larger proteins, it seems to grow quadratically. There are a few outlier proteins whose running time is significantly longer than for proteins of similar size.

Indeed, the running time strongly depends on how often the Viterbi algorithm is executed in the recursion and how quickly a big, non-rigid graph is segmented into several subgraphs. The worst scenario occurs when many Viterbi calculations are required for a protein with densely connected protein graph and with a high degree of flexibility such as intrinsically disordered proteins [26]. In these problematic cases, the signal derived from the mean-variance metric fails to distinguish the labels of inter/intra vertices and edges in the line graph.

Other segmentation methods and ours all require 3D protein structures which are not always available. In our graph-based framework, we may resolve this shortcoming by estimating a protein graph as follows. First, from a given protein primary sequence, we may use its protein contact map predicted, for example, by AlphaFold [27] to construct a protein graph. Second, due to the absence of 3D protein structures, the rigidity estimation could not base on RMSD but rather on another quantity which could be inferred directly from the protein contact map. Final, the rest of the graph-based method is unchanged and still applicable with above predicted protein graph.

Conclusion

We present a new algorithm to characterize structural transitions in proteins. Our graph-based algorithm constructs a graph from a set of protein conformations and detects rigid domains via an edge labeling strategy. A key feature is that the number of rigid domains is determined automatically. Yet the algorithm allows users to relax the rigidity definition of domains and thereby increase or decrease the number of rigid domains. Segmentations produced by our algorithm agree strongly with segmentations found by other methods such as DynDom [3, 28] and Spectrus [6] on various medium to large scale structural transitions.

Our approach has several advantages over other rigid segmentation methods. First, there is no limitation on the number of protein conformations. In fact, a larger number of conformations should result in a better signal and thereby a superior performance of the algorithm. Second, by using the graph-based model along with a binary labeling of edges, we overcome the need to choose the number of rigid domains, which is necessary for many of the existing methods. Moreover, our method performs well with default parameter settings, which saves the user from parameter tweaking. Another appealing aspect of our method is that it can be used to produce a good initial segmentation for other segmentation algorithms. For instance, the *Nguyen*&*Habeck*2016 method [8] requires a good initial guess of the rigid-domain segmentation which could be provided by our graph-based method. Finally, our graph-based framework is quite flexible in that

it allows us to integrate into the scoring function additional information such as the location of hinges or a prior segmentation.

Methods

We organize the "Methods" section as follows. First, we present the notation used throught the Methods section. Next, we describe several steps in our approach such as the coarse-graining algorithm used to reduce the graph size, a line graph transformation that enables inference of edges' labels , and an outlier-detection method that we use to define features on the line graph. Moreover, we explain our method from the perspective of conditional random fields (CRFs) as well as our objective function for labellings of the line graph. Finally, we present pseudo code for our algorithm as well as a post-processing procedure.

Notation

Our algorithm aims to infer a rigid-domain segmentation from $M > 1$ conformational states of a protein. Each conformational state is encoded by a $N \times 3$ matrix $X \in \mathbb{R}^{N\times 3}$ whose rows are the 3D coordinates of representative atoms (typically Cα atoms), i.e. $X_n^{(m)}$ is the position of the nth atom in the mth conformation. Every conformational state gives rise to a symmetric $N \times N$ distance matrix $D^{(m)}$:

$$D_{k,l}^{(m)} := \|X_k^{(m)} - X_l^{(m)}\| \quad (k, l = 1, 2, \ldots, N), \tag{1}$$

where $\|\cdot\|$ denotes the Euclidian norm.

We encode the conformational variability across all M structures through a *protein graph*

$$\mathcal{PG} = (\mathcal{V}, \mathcal{E}) \tag{2}$$

whose vertices $\mathcal{V}$ are the representative atoms $\{1, 2, \ldots, N\}$. An edge between atoms k, l belongs to the edge set $\mathcal{E}$ if and only if

$$\max_{m=1,2,\ldots,M} D_{k,l}^{(m)} \leq \delta \tag{3}$$

where δ is a cutoff distance. Viloria et al. [29] suggest a cutoff distance of 5 Å as optimal value for molecular dynamics simulations. In contrast, HingeProt [9] uses 13 Å as a cutoff to construct a network. Our choice of the cutoff distance is inspired by elastic network models [30], which also encode protein structures as graphs. We ran tests with various cutoff values $\delta = 7.5$, 10.5 and 13.5 Å. We assess the rigidity of a subset $\mathcal{S} \subseteq \mathcal{V}$ through

$$\mathrm{RMSD}(\mathcal{S}) := \frac{2}{M(M-1)} \sum_{m=1}^{M-1} \sum_{m'=m+1}^{M} \mathrm{RMSD}_{\mathcal{S}}\left(X^{(m)}, X^{(m')}\right) \tag{4}$$

where $\mathrm{RMSD}_{\mathcal{S}}\left(X^{(m)}, X^{(m')}\right)$ is the root mean square deviation (RMSD) [31] between conformations $X^{(m)}$ and $X^{(m')}$ reduced to atoms in $\mathcal{S}$. A subset $\mathcal{S}$ is rigid if and only if $\mathrm{RMSD}(\mathcal{S}) < \theta$. The rigidity threshold θ depends on the heterogeneity of the

conformational states. RigidFinder [5] probes every cutoff between 1.0 and 6.0 Å. We typically set $\theta = 3.5$ Å in our tests on the DynDom benchmark [28].

Coarse graining of the protein graph

Rigid domains form densely connected subsets of nodes in the protein graph. To reduce the size of the protein graph, we run the Louvain algorithm [13, 32, 33] that partitions the nodes $\mathcal{V}$ into communities. The parameters of the Louvain algorithm are chosen such that the communities

- are small enough to include, with a few exceptions, amino acids that are part of the same rigid domain (i.e. criterion (Eq. 4) is met for every community);
- are large enough to enable the inference of vertex labels (Eq. 9).

If $\mathcal{C}$ is a partition found by the Louvain algorithm, the *coarse-grained graph*

$$\mathcal{CG} = (\mathcal{CV}, \mathcal{CE}) \tag{5}$$

links two communities c_1 and c_2 ($c_1, c_2 \in \mathcal{C}$) by an undirected edge $(c_1, c_2) \in \mathcal{CE}$ if at least one pair of amino acids $a_1 \in c_1, a_2 \in c_2$ is linked in the protein graph: $(a_1, a_2) \in \mathcal{E}$. In this context, we use the expressions "vertex in the coarse-grained graph" and "community" interchangeably.

The mean variance of all distances between two communities c_1 and c_2 is defined by

$$\begin{aligned}\xi_D(c_1, c_2) :=& \frac{1}{|c_1||c_2|(M-1)} \times \\ & \times \sum_{a_1 \in c_1} \sum_{a_2 \in c_2} \sum_{m=1}^{M} \left(D^{(m)}_{a_1,a_2} - \frac{1}{M} \sum_{m'=1}^{M} D^{(m')}_{a_1,a_2} \right)^2 .\end{aligned} \tag{6}$$

The mean variance is a key quantity of our method. For better readability we skip the subscript when it does not lead to misunderstandings.

We also use RMSD($\mathcal{CG}$) to denote the root mean square deviation calculated from the protein graph of $\mathcal{CG}$ according to Eq. (4).

Line graph transformation

Given an undirected graph with defined sets of vertices and edges, its line graph transformation is a graph whose vertices are the edges in the original graph [34]. Two vertices in the line graph are linked if and only if their corresponding edges in the original graph are incident (share a common vertex).

In this study, we apply the line graph transformation to the coarse-grained graph with a small modification. This transformation is an *intermediate* step that allows us to utilize the generalized Viterbi algorithm to infer binary labels of edges in the coarse-grained graph. The line graph derived from the coarse-grained graph is denoted as:

$$\mathcal{LG}(\mathcal{CG}) = (\mathcal{LV}, \mathcal{LE}) \tag{7}$$

where the edges of the coarse-grained graph become the nodes of the line graph, or $\mathcal{LV} = \mathcal{CE}$. Two vertices are linked if and only if their two corresponding edges in the

coarse-grained graph are incident and the two end nodes are not connected. Formally, we denote two adjacent vertices $v_1 = (c_0, c_1)$ and $v_2 = (c_0, c_2)$ where $v_1, v_2 \in \mathcal{LV}$, and $c_0, c_1, c_2 \in \mathcal{CV}$. In this notation, we call c_0 as a common vertex/node between v_1 and v_2, while c_1, c_2 are end nodes. We create an edge $e = (v_1, v_2) \in \mathcal{LE}$ if and only if c_0 is a common node and $(c_1, c_2) \notin \mathcal{CE}$.

Additionally, we define the mean variance of a vertex v in the line graph $\xi(v)$ according to Eq. (6) evaluated on both communities linked by v. Similarly, the mean variance of an edge e in the line graph is denoted by $\xi(e)$ and defined via the same equation applied to the end nodes of e.

Outlier detection

The bigger the mean variance of a line graph vertex, the more likely is it that the corresponding communities belong to two different domains. Likewise, the end nodes of an edge tend to belong to different domains if the mean variance is large. However, it is not obvious how to define a mapping that is valid across a diverse set of proteins.

Motivated by these observations, we denote by an *inter/intra* vertex a line graph node linking two communities that are part of different domains/the same domain, respectively. Similarly, a line graph edged is an *inter* edge if its end nodes belong to different rigid domains; otherwise it is an intra-domain edge. We note that the mean variance of inter/intra vertices or edges follow two different but overlapping distributions. Both distributions can be modeled with inverse gamma distributions whose parameters can be estimated with expectation maximization (EM). However, we obtained very poor results with this approach due to the small number of inter vertices/edges. Therefore, we only consider the distribution of values from intra vertices/edges and treat values of inter vertices/edges as outliers.

To identify outliers, we use the algorithm developed by [35] that detects outliers based on the distance from its median normalized by the median absolute deviation (*MAD*) [36]. MAD is a measure of dispersion estimated via the median of absolute deviations from the median of the data. We consider a line graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with P vertices $v_i \in \mathcal{V}$ $(i = 1 \dots P)$ and Q edges $e_j \in \mathcal{E}$ $(j = 1 \dots Q)$. Without loss of generality, we enumerate the line graph vertices such that elements in the array of mean variances $\mathcal{A}_{vertex} = [\xi(v_1), \xi(v_2), \dots, \xi(v_P)]$ are sorted in ascending order. Correspondingly, $\mathcal{A}_{edge} = \left[\xi(e_1), \xi(e_2), \dots, \xi(e_Q)\right]$ is the array of mean variances of all edges indexed such that their mean variance increases. For both arrays, we define a binary outlier indicator $\gamma \in \{-1, +1\}$:

$$\gamma(v|\mathcal{A}_{vertex}) = \gamma_v := \begin{cases} -1 & \text{if } v \text{ is an outlier in } \mathcal{A}_{vertex}; \\ +1 & \text{otherwise.} \end{cases}$$

and

$$\gamma\left(e|\mathcal{A}_{edge}\right) = \gamma_e := \begin{cases} -1 & \text{if } e \text{ is an outlier in } \mathcal{A}_{edge}; \\ +1 & \text{otherwise.} \end{cases}$$

When the ascending mean variance arrays of vertices and edges are unambiguous in the given context, we omit the array and indicate whether we are considering vertex or edge arrays by the subscript.

Outliers are characterized by a mean variance that is larger than any other mean variance. The set of outliers can be enlarged by including non-outliers located at the end of the array. By such expanding, it is important to notice that the indices of outliers are always bigger than ones of non-outliers.

A short introduction into CRFs

Let us consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose nodes we call *sites* and $\mathcal{V} = \{1, 2, \ldots, N\}$ without loss of generality. Sites are labeled by elements of the finite set $\mathcal{B}$. Words of length ℓ over the finite alphabet $\mathcal{O}$ are called *observations*. $\mathcal{E}$ is the set of edges in the site graph $\mathcal{G}$. The neighborhood $\mathcal{N}_i \subseteq \mathcal{V}$ of site $i \in \mathcal{V}$ consists of all sites $j \in \mathcal{V}, j \neq i$ that are linked to i by an edge in $\mathcal{N}$ and $i \notin \mathcal{N}_i$. For every label sequence $\boldsymbol{y} \in \mathcal{B}^N$ and subset $I \subseteq \mathcal{V}$, $\boldsymbol{y}_I$ denotes the partial labeling of sites in I: $\boldsymbol{y}_I := \{(i, y_i) \mid i \in I\}$. Additionally, for every $e \in \mathcal{E}$, $\boldsymbol{y}_e$ denotes the labels of two vertices of e and $\boldsymbol{y}_{\mathcal{G}'}$ is the labels of all vertices in a graph $\mathcal{G}'$.

A pair $(\boldsymbol{X}, \boldsymbol{Y})$ composed of a random observation $\boldsymbol{X} \in \mathcal{O}^N$ and a random label sequence $\boldsymbol{Y} \in \mathcal{B}^N$ realizes a feature-based exponential model if the conditional probability $\mathrm{p}(\boldsymbol{y}|\boldsymbol{x})$ of all pairs $(\boldsymbol{x}, \boldsymbol{y})$ is

$$\mathrm{p}(\boldsymbol{y}|\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \exp\left(\sum_{s=1}^{c} \sum_{|I|=s} \Psi^{(s)}(\boldsymbol{y}_I, \boldsymbol{x})\right), \tag{8}$$

where

$$Z(\boldsymbol{x}) := \sum_{\boldsymbol{y}' \in \mathcal{B}^N} \exp\left(\sum_{s=1}^{c} \sum_{|I|=s} \Psi^{(s)}(\boldsymbol{y}'_I, \boldsymbol{x})\right).$$

$\sum_{|I|=s}$ denotes a sum over all cliques I of size s in $\mathcal{G}$; c is the maximum clique size. For every clique size $s \leq c$, the function $\Psi^{(s)}(\boldsymbol{y}_I, \boldsymbol{x})$ is the feature of cliques of size s. Under very weak assumptions the feature-based exponential models coincide with the class of *conditional random fields* where at every site i the label is conditionally independent of the labels outside $\mathcal{N}_i$ given the observation and the labels of $\mathcal{N}_i$.

The labeling problem is solved by computing a labeling sequence

$$\boldsymbol{y}^* := \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{B}^N} \mathrm{p}(\boldsymbol{y}|\boldsymbol{x}) \tag{9}$$

that achieves maximum posterior probability (MAP prediction). In general, MAP prediction is **NP**-hard. The generalized Viterbi algorithm detailed in [14] is able to make the inference for an arbitrary graph, yet has an exponential running time according to the boundary set of a graph. Only if the underlying site graph is small enough, it can be used within a feasible time bound.

Label inference via the generalized Viterbi algorithm

A shortcoming of existing rigid-domain detection methods such as [6, 8, 28] is the requirement to specify the number of rigid domains which is often unknown. To overcome this issue, we use the generalized Viterbi algorithm to infer a binary labeling which indicates if a pair of nodes in the coarse-grained graph belongs to identical or different

rigid domains. It is important to note that we need to infer the binary labels of *edges* in the coarse-grained graph, whereas the Viterbi algorithm estimates optimal *vertex* labels. Thus, it is not suitable to directly apply the Viterbi algorithm to the coarse-grained graph. Instead, we apply the generalized Viterbi algorithm on a line graph derived from the coarse-grained graph. This gives us a binary labeling of line graph vertices, which equivalent to a binary labeling of edges in the coarse-grained graph.

Thus, we consider a line graph as a site graph described above. In a pairwise CRF, one only considers cliques formed by vertices and edges. Consequently, Eq. (8) can be rewritten as

$$\mathrm{p}\,(\boldsymbol{y}|\mathcal{V},\mathcal{E}\,) \sim \exp\left(\sum_{v\in\mathcal{V}} \Psi^{(1)}(v,\boldsymbol{y}_v) + \sum_{(v_1,v_2)\in\mathcal{E}} \Psi^{(2)}(v_1,v_2,\boldsymbol{y}_{(v_1,v_2)})\right) \tag{10}$$

where $\Psi^{(1)}$ and $\Psi^{(2)}$ are the feature functions defined on vertices and edges respectively. The term $Z(\boldsymbol{x})$ can be ignored because it is not a function of $\boldsymbol{y}$. As a convention, we call $\mathrm{p}\,(\boldsymbol{y}|\mathcal{V},\mathcal{E}\,)$ "*unnormalized probability*" or "*scoring function*" interchangeably.

In our rigid domains detection problem, we define a feature function for a vertex v along with its label $\boldsymbol{y}_v$ by

$$\Psi^{(1)}(v,\boldsymbol{y}_v) = \gamma_v \boldsymbol{y}_v\,. \tag{11}$$

This function will reward labeling $\boldsymbol{y}_v$ that coincide with the outlier indicator value.

Given an edge $e = (v_1, v_2) \in \mathcal{E}$, we define a feature function on e and its predicted label $\boldsymbol{y}_e$ by distinguishing three cases:

Case "Two values among $\gamma_e, \gamma_{v_1}, \gamma_{v_2}$ are equal to -1." In this case, the egde feature rewards an agreement between the predicted vertex labels $\boldsymbol{y}_e$ and the outlier indicators:

$$\Psi^{(2)}\big(e,\boldsymbol{y}_e|e=(v_1,v_2)\big) := \begin{cases} +1 & \text{if } \boldsymbol{y}_{v_1}\gamma_{v_1} + \boldsymbol{y}_{v_2}\gamma v_2 = 2; \\ -1 & \text{otherwise.} \end{cases} \tag{12}$$

Case "$\gamma_{v_1} = \gamma_{v_2} = +1$" seems to indicate that three nodes of v_1 and v_2 (a common vertex and two end nodes) belong to the same rigid component. However, the vertex shared by the two edges may be part of a hinge region between two rigid components. This is likely to occur if the mean variance value of the edge is outlier, or "$\gamma_e = -1$". If this is the case, we have to decide to which component the hinge node belongs. This decision is based on a comparison between $\xi(v_1)$ and $\xi(v_2)$. Thus, $\Psi^{(2)}$ becomes:

$$\begin{aligned} &\Psi^{(2)}\big(e,\boldsymbol{y}_e|e=(v_1,v_2)\big) \\ &\quad := \begin{cases} +1 & \text{if } \boldsymbol{y}_{v_1} = -1, \boldsymbol{y}_{v_2} = +1, \gamma_e = -1 \text{ and } \xi_{v_1} > \xi_{v_2}; \\ +1 & \text{if } \boldsymbol{y}_{v_1} = +1, \boldsymbol{y}_{v_2} = -1, \gamma_e = -1 \text{ and } \xi_{v_1} < \xi_{v_2}; \\ +1 & \text{if } \boldsymbol{y}_{v_1} = \boldsymbol{y}_{v_2} = +1 \text{ and } \gamma_e = +1; \\ 0 & \text{if } \boldsymbol{y}_{v_1}\boldsymbol{y}_{v_2} = -1, \gamma_e = -1 \text{ and } \xi_{v_1} = \xi_{v_2}; \\ -1 & \text{otherwise.} \end{cases} \end{aligned} \tag{13}$$

For any other combination of $\gamma_{v_1}, \gamma_{v_2}$ and γ_e, we set

$$\Psi^{(2)}\big(e,\boldsymbol{y}_e\big) := 0 \tag{14}$$

In all three cases above, labelings are rewarded by setting $\Psi^{(2)}$ to $+1$, penalized by setting $\Psi^{(2)}$ to -1 and ignored by setting $\Psi^{(2)}$ to 0.

Hence, for any labeling of the line graph $\mathcal{G}$, the generalized Viterbi algorithm computes its unnormalized probability (Eq. 10) via Eqs. (11)–(14) and thus gives us the most probable labels of $\mathcal{G}$.

Graph-based prediction of rigid domains

This subsection provides pseudo code for our graph-based prediction of rigid domains in proteins. We denote the rigidity threshold as θ (typically 3.5 Å).

$\theta \leftarrow$ 3.5
$\mathcal{PG} = (\mathcal{PV}, \mathcal{PE}) \leftarrow X^{(m)}, m = 1..M$: a protein graph is constructed from multiple conformations.
$\mathcal{CG} = (\mathcal{CV}, \mathcal{CE}) \leftarrow$ a coarse-grained graph is obtained by running the Louvain algorithm on $\mathcal{PG}$
function SEGMENT($\mathcal{CG}$)
  $Final\_List \leftarrow$ []: an empty list containing segmented subgraphs
  if RMSD$(\mathcal{CG}) > \theta$ **then**
    $\mathcal{LG} = (\mathcal{LV}, \mathcal{LE}) \leftarrow \mathcal{CG}$: a line graph is constructed from the coarse-grained graph
    $\boldsymbol{y}_{\mathcal{LG}} \leftarrow$ Viterbi$(\mathcal{LG})$: the most probable labels of the line graph $\mathcal{LG}$ are computed with the gneralized Viterbi algorithm
    $\boldsymbol{y}_{e:e\in\mathcal{CG}} \leftarrow \boldsymbol{y}_{\mathcal{LG}}$: trace back labels of edges in the coarse-grained graph from the labels of vertices in the line graph
    $[\mathcal{CG}_1, .., \mathcal{CG}_K] \leftarrow \boldsymbol{y}_{e:e\in\mathcal{CG}}$: obtain K disconnected subgraphs by removing negative labels of edges in the coarse-grained graph $\mathcal{CG}$
    if K > 1 **then**
      for each $g \in [\mathcal{CG}_1, .., \mathcal{CG}_K]$ **do**
        if RMSD$(g) > \theta$ **then**
          Run **Segment**(g)
        else
          $Final\_List$.add(g)
        end if
      end for
    else
      if $\mathcal{CG}_1 \neq \mathcal{CG}$ **then**
        Run **Segment**$(\mathcal{CG}_1)$
      else
        Relax the outlier detection by enlarging the set of outliers toward the top 5% biggest values of non-outliers
        Run **Segment**$(\mathcal{CG})$ with the new set of outliers
      end if
    end if
  else
    return $\mathcal{CG}$
  end if
  return $Final\_List$
end function

There is no guarantee that this algorithm always converges. However, we experienced fast convergence within a few iterations in most of our experiments. We also added a limitation on the number of recursions. The final result of our algorithm is a list of disconnected subgraphs of the coarse-grained graph.

Finalizing rigid-domain segmentation

Our graph-based method for rigid-domain detection described in the Sect. 5.6 produces a list of disconnected subgraphs of the reduced graph. we can trace back the subgraphs to the corresponding protein subgraphs and thus obtain a list of disconnected protein graphs.

Let $\mathcal{S} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_L\}$ be a mutual exclusive partition of the protein graph $\mathcal{PG}$. Our merging algorithm works as follows:

$$
\begin{aligned}
&\text{Merging Algorithm}\\
&\quad val \leftarrow \max_{\mathcal{S}_i,\mathcal{S}_j \in \mathcal{S}} \frac{\mathrm{RMSD}\left(\mathcal{S}_i\right)+\mathrm{RMSD}\left(\mathcal{S}_j\right)}{\mathrm{RMSD}\left(\mathcal{S}_i \cup \mathcal{S}_j\right)}\\
&\quad \textbf{while } val > 1 \textbf{ do}\\
&\qquad \{\mathcal{D}_*, \mathcal{D}'_*\} \leftarrow \operatorname*{argmax}_{\{\mathcal{D},\mathcal{D}'\} \in \mathcal{S}} \frac{\mathrm{RMSD}\left(\mathcal{D}\right)+\mathrm{RMSD}\left(\mathcal{D}'\right)}{\mathrm{RMSD}\left(\mathcal{D} \cup \mathcal{D}'\right)}\\
&\qquad \mathcal{D}_* \leftarrow \mathcal{D}_* \cup \mathcal{D}'_*\\
&\qquad \text{Remove } \mathcal{D}'_* \text{ from } \mathcal{S}\\
&\qquad val \leftarrow \max_{\{\mathcal{D},\mathcal{D}'\} \in \mathcal{S}} \frac{\mathrm{RMSD}\left(\mathcal{D}\right)+\mathrm{RMSD}\left(\mathcal{D}'\right)}{\mathrm{RMSD}\left(\mathcal{D} \cup \mathcal{D}'\right)}
\end{aligned}
$$

After termination of the Merging Algorithm, $\mathcal{S}$ is returned as rigid-domain prediction.

Supplementary information

The online version contains supplementary material available at https://doi.org/10.1186/s12859-021-03966-3.

Additional file 1: The support information includes the follows. The first section is the analyses of inconsistency error of reduced graph. The second section is the analyses of signal enhancement by coarse graining. The third section contains PDB codes of Lysozyme protein. The fourth section is the analyses of the running time.

Acknowledgements
Linh Dang and Mehmet Gueltas thank Felix Heinrich for his help to create the web server. Stephan Waack and Linh Dang acknowledge support by the Open Access Publication Funds of the Göttingen University.

Authors' contributions
The model was designed by LD and SW with advice from MH. The algorithm was designed and implemented by LD. The analysis and evaluation were performed by LD and TN. The manuscript was written by all authors and revised by MH and LD. A web server is developed by MG. SW and LD conduct the study. All authors read and approved the final manuscript.

Funding
Open Access funding enabled and organized by Projekt DEAL. Michael Habeck and Thach Nguyen haven been supported by Deutsche Forschungsgemeinschaft (DFG), SFB 860 TP B09. The funding body did not play any roles in the design of the study and collection, analysis, and interpretation of data and in writing the manuscript.

Availability of data and materials
Web-server: http://azifi.tz.agrar.uni-goettingen.de/webservice/ Source code: https://github.com/dtklinh/GBRDE.

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Institute of Computer Science, University of Göttingen, Goldschmidtstr 7, 37077 Göttingen, Germany. [2] Felix Bernstein Institute for Mathematical Statistics in the Biosciences, University of Göttingen, Goldschmidtstr 7, 37077 Göttingen, Germany. [3] Max Planck Institute for Biophysical Chemistry, Am Fassberg 11, 37077 Göttingen, Germany. [4] Microscopic Image Analysis Group, University Hospital Jena, Am Klinikum 1, 07747 Jena, Germany. [5] Breeding Informatics Group, Department of Animal Sciences, Margarethe von Wrangell-Weg 7, 37075 Göttingen, Germany. [6] Center for Integrated Breeding Research (CiBreed), Albrecht-Thaer-Weg 3, 37075 Göttingen, Germany.

Received: 28 May 2020 Accepted: 8 January 2021
Published online: 12 February 2021

References

1. Henzler-Wildman K, Kern D. Dynamic personalities of proteins. Nature. 2007;450:964–72.
2. Gerstein M, Lesk AM, Chothia C. Structural mechanisms for domain movements in proteins. Biochemistry. 1994;33(22):6739–49.
3. Hayward S, Berendsen HJ. Systematic analysis of domain motions in proteins from conformational change: new results on citrate synthase and t 4 lysozyme. Proteins Struct Funct Genet. 1998;30(2):144–54.
4. Wriggers W, Schulten K. Protein domain movements: detection of rigid domains and visualization of hinges in comparisons of atomic coordinates. Proteins Struct Funct Genet. 1997;29(1):1–14.
5. Abyzov A, Bjornson R, Felipe M, Gerstein M. Rigidfinder: a fast and sensitive method to detect rigid blocks in large macromolecular complexes. Proteins Struct Funct Bioinform. 2010;78(2):309–24.
6. Ponzoni L, Polles G, Carnevale V, Micheletti C. Spectrus: a dimensionality reduction approach for identifying dynamical domains in protein complexes from limited structural datasets. Structure. 2015;23(8):1516–25.
7. Hirsch M, Habeck M. Mixture models for protein structure ensembles. Bioinformatics. 2008;24(19):2184–92.
8. Nguyen T, Habeck M. A probabilistic model for detecting rigid domains in protein structures. Bioinformatics. 2016;32(17):710–7.
9. Emekli U, Schneidman-Duhovny D, Wolfson HJ, Nussinov R, Haliloglu T. Hingeprot: automated prediction of hinges in protein structures. Proteins Struct Funct Bioinform. 2008;70(4):1219–27.
10. Hinsen K. Analysis of domain motions by approximate normal mode calculations. Proteins Struct Funct Bioinform. 1998;33(3):417–29.
11. Flores SC, Gerstein MB. Flexoracle: predicting flexible hinges by identification of stable domains. BMC Bioinform. 2007;8(1):215.
12. Mueller CW, Schlauderer GJ, Reinstein J, Schulz GE. Adenylate kinase motions during catalysis: an energetic counterweight balancing substrate binding. Structure. 1996;4:147–56.
13. Traag VA, Van Dooren P, Nesterov Y. Narrow scope for resolution-limit-free community detection. Phys Rev E. 2011;84(1):016114. https://doi.org/10.1103/PhysRevE.84.016114.
14. Dong Z, Wang K, Dang TKL, Gültas M, Welter M, Wierschin T, Stanke M, Waack S. Crf-based models of protein surfaces improve protein-protein interaction site predictions. BMC Bioinform. 2014;15:277. https://doi.org/10.1186/1471-2105-15-277.
15. Whitford PC, Miyashita O, Levy Y, Onuchic JN. Conformational transitions of adenylate kinase: switching by cracking. J Mol Biol. 2007;366:1661–71.
16. Habeck M, Nguyen T. A probabilistic network model for structural transitions in biomolecules. Proteins. 2018;86:634–43. https://doi.org/10.1002/prot.25490.
17. Lee RA, Razaz M, Hayward S. The DynDom database of protein domain motions. Bioinformatics. 2003;19(10):1290–1.
18. Lim K, Read RJ, Chen CCH, Tempczyk A, Wei M, Ye D, Wu C, Dunaway-Mariano D, Herzberg O. Swiveling domain mechanism in pyruvate phosphate dikinase. Biochemistry. 2007;46(51):14845–53. https://doi.org/10.1021/bi701848w.
19. Theis K, Gong P, Martin CT. Topological and conformational analysis of the initiation and elongation complex of T7 RNA polymerase suggests a new twist. Biochemistry. 2004;43(40):12709–15. https://doi.org/10.1021/bi0486987.
20. Boisvert DC, Wang J, Otwinowski Z, Norwich AL, Sigler PB. The 2.4 å crystal structure of the bacterial chaperonin GroEL complexed with atpgs. Nat Struct Biol. 1996;3(2):170–7. https://doi.org/10.1038/nsb0296-170.
21. Karmen A, Wroblewski F, Ladue JS. Transaminase activity in human blood. J Clin Investig. 1955;34(1):126–31. https://doi.org/10.1172/JCI103055.
22. Blake CCF, Koenig DF, Mair GA, North ACT, Phillips DC, Sarma VR. Structure of hen egg-white lysozyme: a three-dimensional fourier synthesis at 2 å resolution. Nature. 1965;206(4986):757–61. https://doi.org/10.1038/206757a0.
23. Zheng Y, Han GW, Abagyan R, Wu B, Stevens RC, Cherezov V, Kufareva I, Handel TM. Structure of CC chemokine receptor 5 with a potent chemokine antagonist reveals mechanisms of chemokine recognition and molecular mimicry by HIV. Immunity. 2017;46(6):1005–10175. https://doi.org/10.1016/j.immuni.2017.05.002.
24. Sievers F, Wilm A, Dineen D, Gibson TJ, Karplus K, Li W, Lopez R, McWilliam H, Remmert M, Söding J, Thompson JD, Higgins DG. Fast, scalable generation of high-quality protein multiple sequence alignments using clustal omega. Mol Syst Biol. 2011;7(1):539. https://doi.org/10.1038/msb.2011.75.
25. Sievers F, Higgins DG. Clustal omega for making accurate alignments of many protein sequences. Protein Sci. 2017;27(1):135–45. https://doi.org/10.1002/pro.3290.
26. Dunker AK, Lawson JD, Brown CJ, Williams RM, Romero P, Oh JS, Oldfield CJ, Campen AM, Ratliff CM, Hipps KW, Ausio J, Nissen MS, Reeves R, Kang C, Kissinger CR, Bailey RW, Griswold MD, Chiu W, Garner EC, Obradovic Z. Intrinsically disordered protein. J Mol Graph Model. 2001;19(1):26–59. https://doi.org/10.1016/s1093-3263(00)00138-8.
27. Senior AW, Evans R, Jumper J, Kirkpatrick J, Sifre L, Green T, Qin C, Žídek A, Nelson AWR, Bridgland A, Penedones H, Petersen S, Simonyan K, Crossan S, Kohli P, Jones DT, Silver D, Kavukcuoglu K, Hassabis D. Improved protein structure prediction using potentials from deep learning. Nature. 2020;577(7792):706–10. https://doi.org/10.1038/s41586-019-1923-7.
28. Hayward S, Kitao A, Berendsen HJC. Model-free methods of analyzing domain motions in proteins from simulation: a comparison of normal mode analysis and molecular dynamics simulation of lysozyme. Proteins Struct Funct Bioinform. 1997;27(3):425–37. https://doi.org/10.1002/(SICI)1097-0134(199703)27:3<425::AID-PROT10>3.0.CO;2-N.
29. Salamanca Viloria J, Allega MF, Lambrughi M, Papaleo E. An optimal distance cutoff for contact-based protein structure networks using side-chain centers of mass. Sci Rep. 2017;7(1):2838. https://doi.org/10.1038/s41598-017-01498-6.

30. Bahar I, Atilgan AR, Erman B. Direct evaluation of thermal fluctuations in proteins using a single-parameter harmonic potential. Fold Des. 1997;2(3):173–81. https://doi.org/10.1016/s1359-0278(97)00024-2.
31. Kabsch W. A solution for the best rotation to relate two sets of vectors. Acta Crystallogr Sect A. 1976;32:922–3. https://doi.org/10.1107/S0567739476001873.
32. Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. J Stat Mech Theory Exp. 2008;2008(10):10008. https://doi.org/10.1088/1742-5468/2008/10/p10008.
33. Traag VA. Faster unfolding of communities: speeding up the Louvain algorithm. Phys Rev E. 2015;92(3):032801. https://doi.org/10.1103/PhysRevE.92.032801. arXiv:1503.01322D.
34. Evans TS, Lambiotte R. Line graphs of weighted networks for overlapping communities. Eur Phys J B. 2010;77(2):265–72. https://doi.org/10.1140/epjb/e2010-00261-8.
35. Iglewicz B, Hoaglin DC. How to detect and handle outliers, vol. 16. Milwaukee: ASQ Press; 1993.
36. Median absolute deviation, p. 348. Springer, New York (2008). https://doi.org/10.1007/978-0-387-32833-1_261.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

1 Inconsistency error of reduced graph

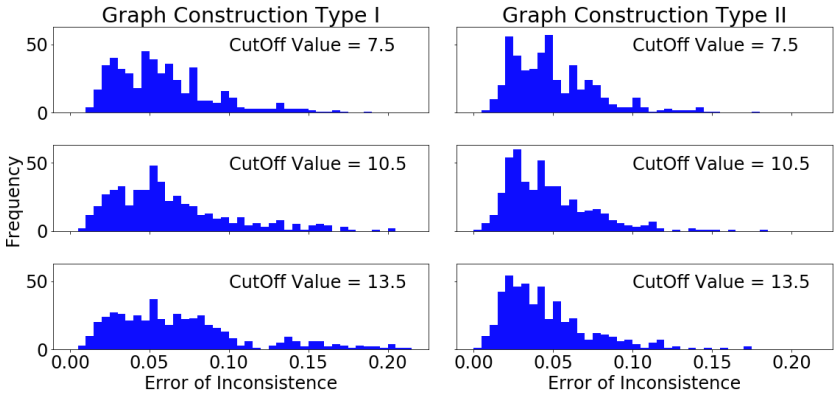

Figure S1: Histogram of inconsistency error from graph construction type I and II and their varying cutoff values respectively.

2 Signal enhancement by coarse graining

As in the Methods and Discussion sections, let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph and $\mathcal{LG} = (\mathcal{LV}, \mathcal{LE})$ the line graph derived from $\mathcal{G}$. For each vertex $v \equiv (v_0, v_1) \in \mathcal{LV}$ $(v_0, v_1 \in \mathcal{V})$, $\xi(v) = \xi(v_0, v_1)$ is the mean-variance of v according to the Equation (6) in the Methods section. The label of v is $\sigma_v = +1$ (intra-vertex) if $\sigma_{v_0} = \sigma_{v_1}$ and -1 (inter-vertex) otherwise. For each edge $e \equiv (v_l, v_m, v_r) \in \mathcal{LE}$ $(v_l, v_m, v_r \in \mathcal{V}; (v_l, v_m), (v_m, v_r) \in \mathcal{E}$ & $(v_l, v_r) \notin \mathcal{E})$, its mean-variance $\xi(e) \equiv \xi(v_l, v_r)$ is calculated as above. The label of an edge in the line graph is $\sigma_{e^*} = +1$ (intra-edge) if $\sigma_{v_l} = \sigma_{v_r}$, and -1 (inter-edge) otherwise.

The performance of the CRF scoring function depends on how well the values of mean-variance of inter and intra-vertices/edges separate in the line graph. We assess this separability by the mean of the area under the ROC curve (AUC) through the line graphs derived from the protein graph and from the coarse-grained graph.

With the thresholding of the mean-variance as a classifier, the Panels S2.A and S2.B show that inter and intra-vertices/edges in the line graph derived from the coarse-grained graph significantly achieve bigger separability than the ones in the line graph derived from the protein graph. Therefore, the coarse-graining process does not only reduce the graph size, but it also enriches the information represented by the mean-variance.

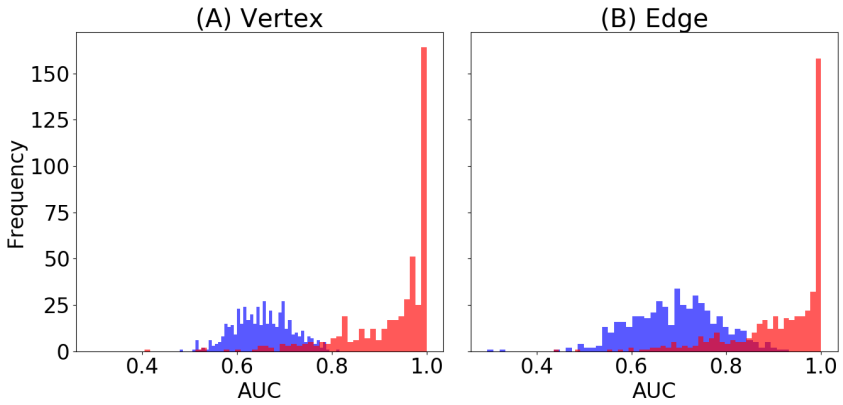

Figure S2: Histograms of area under the ROC curve (AUC) evaluated on 487 proteins in the DynDom dataset. (A) Histograms of AUC calculated from the inter and intra-vertices in the line graphs derived from the protein graph(blue histogram) and from the coarse-grained graph (red histogram). (B) Histograms of AUC calculated from the inter and intra-edges in the line graphs derived from the protein graph(blue histogram) and from the coarse-grained graph (red histogram).

3 Lysozyme protein

PDB codes used in the study

150L_D 173L_A 150L_A 150L_C 150L_B 3SB8_C 3GUN_A 3GUN_B 3GUL_B 3GUL_A 152L_A 2HUK_A 2Q9E_C 4S0W_A 1JQU_D 1JQU_A 1JQU_B 1JQU_C 149L_A 145L_A 151L_A 2HUM_B 2HUM_A 3GUK_A 3GUK_B 137L_B 3FI5_A 3SB9_B 1L97_A 1L97_B 3GUI_A 178L_A 3GUO_A 3GUO_B 201L_B 201L_A 3GUM_B 3GUM_A 168L_B 168L_C 168L_A 168L_D 168L_E 172L_A 3GUJ_A 209L_A 1P7S_A 1KNI_A 1SSY_B 1SSY_A 1QTH_B 4PK0_A 4UIS_G 169L_A 169L_C 169L_B 169L_E 169L_D 4GBR_B 1LWG_A 3GUP_B 167L_B 3GUP_A 2LCB_A 3SB9_A 3SBA_F 3SBA_D 3SBA_E 3SBA_B 3SBA_C 3SBA_A 2QAR_C 2QAR_F 1PQK_C 4PJZ_A 171L_A 3SBB_C 218L_A 3SB7_A 3SB7_B 1QTH_A 3JR6_A 3JR6_B 3JR6_C 3JR6_D 3SB5_B 3SB5_C 3SB5_A 3SB5_D 214L_A 1T8G_A 189L_A 1P5C_A 1P5C_B 1P5C_C 1P5C_D 216L_B 216L_A 174L_B 174L_A

4 Running time

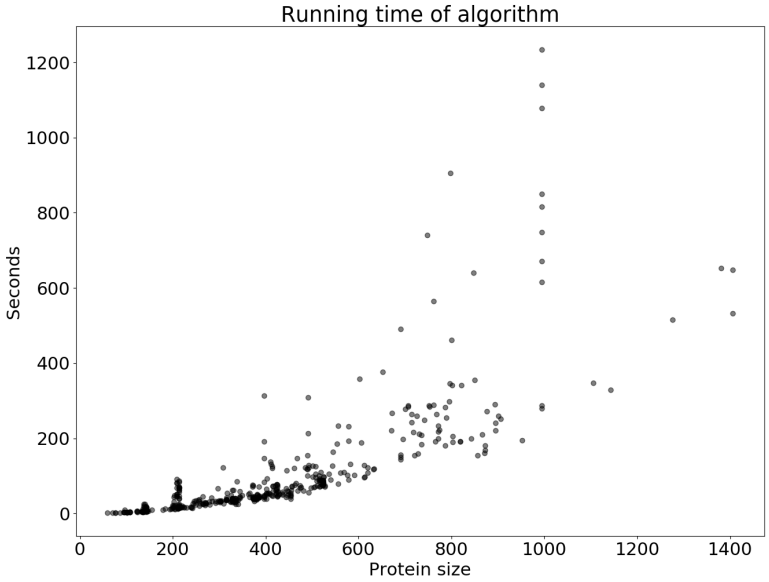

Figure S3: Protein size versus running time (measured in seconds) evaluated for 487 proteins selected from the DynDom database.

A.2. The Prediction of DNA-binding sites in Proteins

entropy

MDPI

Article

A Novel Sequence-Based Feature for the Identification of DNA-Binding Sites in Proteins Using Jensen–Shannon Divergence

Truong Khanh Linh Dang [1], Cornelia Meckbach [2], Rebecca Tacke [2], Stephan Waack [1] and Mehmet Gültas [1,*]

[1] Institute of Computer Science, University of Göttingen, Göttingen 37077, Germany; ldang1@informatik.uni-goettingen.de (T.K.L.D.); waack@informatik.uni-goettingen.de (S.W.)

[2] Institute of Bioinformatics, University Medical Center Göttingen, Göttingen 37077, Germany; c.meckbach@bioinf.med.uni-goettingen.de (C.M.); rebecca.tacke@stud.uni-goettingen.de (R.T.)

* Correspondence: gueltas@informatik.uni-goettingen.de; Tel.: +49-551-39-172055

Academic Editors: Carlos M. Travieso-González and Jesús B. Alonso-Hernández
Received: 30 July 2016; Accepted: 20 October 2016; Published: 24 October 2016

Abstract: The knowledge of protein-DNA interactions is essential to fully understand the molecular activities of life. Many research groups have developed various tools which are either structure- or sequence-based approaches to predict the DNA-binding residues in proteins. The structure-based methods usually achieve good results, but require the knowledge of the 3D structure of protein; while sequence-based methods can be applied to high-throughput of proteins, but require good features. In this study, we present a new information theoretic feature derived from Jensen–Shannon Divergence (JSD) between amino acid distribution of a site and the background distribution of non-binding sites. Our new feature indicates the difference of a certain site from a non-binding site, thus it is informative for detecting binding sites in proteins. We conduct the study with a five-fold cross validation of 263 proteins utilizing the Random Forest classifier. We evaluate the functionality of our new features by combining them with other popular existing features such as position-specific scoring matrix (PSSM), orthogonal binary vector (OBV), and secondary structure (SS). We notice that by adding our features, we can significantly boost the performance of Random Forest classifier, with a clear increment of sensitivity and Matthews correlation coefficient (MCC).

Keywords: entropy; Jensen–Shannon divergence; Random Forest; DNA-binding sites

1. Introduction

Interactions between proteins and DNA play essential roles for controlling of several biological processes such as transcription, translation, DNA replication, and gene regulation [1–3]. An important step to understand the underlying molecular mechanisms of these interactions is the identification of DNA-binding residues in proteins. These residues can provide a great insight into the protein function which leads to gene expression and could also facilitate the generation of new drugs [4,5].

Until now, several groups have published different studies based on either experimental or computational identification of DNA-binding proteins [1,6–11] as well as residues in these proteins [12–23]. However, the usage of experimental approaches for the determination of binding sites is still challenging since they are often demanding, relatively expensive, and time-consuming. To overcome the difficulty of experimental approaches, it is highly desired to develop fast and reliable computational methods for the prediction of DNA-binding residues. For this purpose, several state-of-the-art prediction methods have been developed for the automated identification of those residues. Such methods can be assigned into two main categories: (i) based on the information observed from structure and sequence in a collective manner; (ii) based on the features derived directly

from the amino acid sequence alone (for more detail see reviews [24] and [25]). Although the first type of approaches provides promising information about DNA-binding residues in proteins, their application is difficult due to the limited number of experimentally determined protein structures. In contrast to structure-based approaches, sequence-based methods have been developed by extracting different sequence information features, like amino acid frequency, position-specific scoring matrix (PSSM), BLOSUM62 matrix, sequence conservation, etc. [3,4,18,19,26,27]. Using these features, several machine learning techniques have been applied to construct the classifiers for the prediction of binding residues in proteins. To this end, a variety of support vector machine (SVM) classifiers have been developed in recent studies [2,17–19,23,26,28]. For example, Westhof et al. have recently used an SVM classifier approach in their study, named RBscore (http://ahsoka.u-strasbg.fr/rbscore/), by using the physicochemical and evolutionary features that are linearly combined with a residue neighboring network [2]. Further, SVM algorithms were also applied for the models proposed in BindN [18], DISIS [19], BindN+ [23], DP-Bind [27] using different sequence information features including the biochemical property of amino acids, sequence conservation, evolutionary information in terms of PSSM, the side chain pKa value, hydrophobicity index, molecular mass and BLOSUM62 matrix. In addition, other machine learning classifiers such as neural network models [13,15], naive Bayes classifier [26], Random Forest classifiers (RF) [4,29,30] have been developed based on the features derived from protein sequences. For example, Wong et al. [29] have recently developed a successful method using RF classifier with both DNA and protein derived features to predict the specific residue-nucleotide interactions for different DNA-binding domain families.

Despite the rich literature on the sequence-based methods as mentioned above, to date there is still a need to find suitable feature extraction approaches that can enhance the characteristics of DNA-binding residues and thus help to improve the performance of existing methods for identification of DNA-binding residues in proteins. For this aim, we introduce and evaluate a new information theory-based method for the prediction of these residues using Jensen–Shannon divergence ($\mathbb{JSD}$). As a divergence measure based on the Shannon entropy, $\mathbb{JSD}$ is a symmetrized and smoothed version of the Kullback-Leibler divergence and is often used for different problems in the field of bioinformatics [31–35]. In this study, following the line of Capra et al. [34] we first quantify the divergence between the observed amino acid distribution of a site in a protein and the background distribution of non-binding sites by using $\mathbb{JSD}$. After that, in analogy to our previous studies QCMF [32] and CMF [36], we incorporate biochemical signals of binding residues in the calculation of $\mathbb{JSD}$ that results in the intensification of the DNA-binding residue signals from the non-binding signals.

To demonstrate the performance and functionality of our proposed approach, we apply Random Forest (RF) classifier using our new $\mathbb{JSD}$ based features together with three widely used machine learning features, namely position-specific scoring matrix (PSSM), secondary structure (SS) information, and orthogonal binary vector (OBV) information (see review [24]). Our results show that using $\mathbb{JSD}$ based features, RF classifier reaches an improved performance in identifying DNA-binding residues with a significantly higher Matthews correlation coefficient (MCC) value in comparison to using previous features alone. Although we only applied RF classifier in this study, both of our sequence-based features could be used in other classifiers such as SVM, neural networks, or decision trees.

2. Results

In this study, we introduce new sequence-based features using $\mathbb{JSD}$ to improve the performance of previous machine learning approaches in identification of DNA-binding residues in proteins. For this purpose, we propose new sequence-based features ($f_{\mathbb{JSD}}$ and $f_{\mathbb{JSD}\text{-t}}$) using $\mathbb{JSD}$ in two different ways. First, using $\mathbb{JSD}$, we calculate the divergences between observed amino acid distributions in multiple sequence alignments (MSAs) of proteins under study and the background distribution which is calculated according to amino acid counts at non-binding residue positions in MSAs. In the second step, we transform the observed amino acid distributions with a doubly stochastic matrix (DSM) to

enhance the weak signal of binding sites in proteins which could not be predicted in the first step. Finally, we calculate for each residue in proteins $\mathbb{JSD}$-based scores and use them for the improvement of the performance of machine learning approaches.

To evaluate our new features, we use two frequently considered cut-off distances of 3.5 Å and 5 Å and thus define a residue in a protein as DNA-binding if the distance between at least one atom on its backbone or side chain and the DNA molecule is smaller than the considered cut-off.

The Results section of this study comprises of two parts. First, we investigate the functionality of our new features combining them in Random Forest (RF) classifier with three previous features. The RF classifier is constructed from 4298 positive and 44,805 negative instances extracted from 263 proteins. The performance of the classifier is evaluated using a five-fold cross validation procedure in which we randomly divided the samples into five parts. The assessment is performed by choosing each of these parts as a test set and the remaining four parts as a training set for model selection. Second, to illustrate the usefulness of our new approach for the prediction of DNA-binding residues, we analyzed the proto-oncogenic transcription factor MYC-MAX (PDB-ID: 1NKP) which is a heterodimer protein complex of two proteins. It is important to note that this protein complex is not included in the training dataset.

2.1. Random Forest Classifier

To apply the Random Forest (RF) classifier, we combine our new features ($f_{\mathbb{JSD}}$ and $f_{\mathbb{JSD}\text{-t}}$) with the features f_{PSSM}, f_{OBV}, and f_{SS} which are widely used for the prediction of DNA-binding residues. Our results show that using our features RF classifier reaches an improved performance in identifying DNA-binding sites with clearly higher statistical values (see Tables 1 and 2). Moreover, we individually evaluated the combination of our features with existing features. The results suggest that the classifier with $f_{\mathbb{JSD}\text{-t}}$ feature has provided better sensitivity and comparable Matthews correlation coefficient (MCC) values in comparison to $f_{\mathbb{JSD}}$ feature. However, its specificity is moderately decreased. A further comparison reveals that the usage of our both features together with other features does not affect the performance of the classifier. The details are presented for 3.5 Å in Table 1 and for 5 Å in Table 2 and in Appendix A with the standard error of each of the performance measures over the values obtained in the five iterations (see Tables A1 and A2).

Table 1. Prediction performance of Random Forest (RF) classifier on different features using a cut-off of 3.5 Å. The prediction system was evaluated by five-fold cross validation.

Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
f_{PSSM}	0.292	0.963	0.307	0.777	0.313
f_{PSSM} + $f_{\mathbb{JSD}}$	0.385	0.949	0.349	0.795	0.369
f_{PSSM} + $f_{\mathbb{JSD}\text{-t}}$	0.41	0.939	0.35	0.802	0.377
f_{PSSM} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	0.414	0.94	0.348	0.800	0.376
f_{PSSM} + f_{SS}	0.339	0.958	0.334	0.794	0.338
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}}$	0.416	0.95	0.378	0.808	0.390
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}\text{-t}}$	0.441	0.94	0.372	0.817	0.401
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	0.439	0.94	0.37	0.814	0.399
f_{PSSM} + f_{OBV} + f_{SS}	0.367	0.968	0.398	0.838	0.413
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}}$	0.422	0.958	0.409	0.837	0.425
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}\text{-t}}$	0.447	0.95	0.403	0.841	0.431
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	0.444	0.947	0.393	0.835	0.423

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

Table 2. Prediction performance of Random Forest (RF) classifier on different features using a cut-off of 5.0 Å. The prediction system was evaluated by five-fold cross validation.

Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
f_{PSSM}	0.286	0.966	0.350	0.778	0.425
$f_{PSSM} + f_{\mathbb{JSD}}$	0.395	0.95	0.407	0.801	0.487
$f_{PSSM} + f_{\mathbb{JSD}\text{-t}}$	0.418	0.943	0.411	0.807	0.494
$f_{PSSM} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.426	0.942	0.414	0.807	0.497
$f_{PSSM} + f_{SS}$	0.334	0.963	0.386	0.796	0.455
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}}$	0.424	0.951	0.436	0.814	0.513
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.448	0.944	0.438	0.820	0.520
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.445	0.944	0.434	0.819	0.521
$f_{PSSM} + f_{OBV} + f_{SS}$	0.337	0.975	0.431	0.830	0.517
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.419	0.958	0.450	0.832	0.535
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.439	0.952	0.453	0.836	0.539
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.442	0.949	0.445	0.832	0.535

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

To further investigate the performance of $\mathbb{JSD}$-based features proposed in this study, we analyzed two additional datasets, namely RBscore [2] and PreDNA datasets [37]. Although the RBscore and PreDNA datasets initially contain 381 and 224 DNA-binding proteins, respectively, we have eliminated a few proteins since they are either included in our training dataset or ineligible due to their MSAs. Consequently, we constructed RF classifier using 263 proteins (which were also used for cross-validation) and randomly selecting 60 proteins from each dataset for testing, respectively. The results of these analyses consistently suggest that our new features show great complementary effect to the previous features which often leads to clear improvement of the classification performance (see Tables 3 and 4). The detailed performance of classifier on different features using different cut-offs for each dataset can be found in Appendix A (see Tables A3–A6).

Considering the AUC-ROC and AUC-PR as the only evaluation factor, results indicate that the RF classifier often achieved its best performance based on both cut-off distances if we combine our new $f_{\mathbb{JSD}\text{-t}}$ feature together with the existing three features (see Tables 1–3). Interestingly, by analyzing the PreDNA dataset we observed that RF classifier with $f_{\mathbb{JSD}}$ or $f_{\mathbb{JSD}\text{-t}}$ features for the cut-off of 3.5 Å showed similar performance. However, regarding to the distance cut-off of 5 Å, the classifier with $f_{\mathbb{JSD}}$ feature reached slightly better performance than those with $f_{\mathbb{JSD}\text{-t}}$ feature (see Table 4). After looking at the overall performances, it is inferred that adding our new features can boost the performance of the RF classifier in terms of AUC-ROC and AUC-PR.

Table 3. Prediction performance of Random Forest (RF) classifier on RBscore dataset using different distance cut-offs.

Cut-Off	Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
3.5 Å	$f_{PSSM} + f_{OBV} + f_{SS}$	0.517	0.976	0.534	0.896	0.528
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.58	0.967	0.54	0.907	0.543
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.612	0.963	0.546	0.910	0.551
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.601	0.962	0.531	0.909	0.546
5.0 Å	$f_{PSSM} + f_{OBV} + f_{SS}$	0.499	0.98	0.584	0.895	0.641
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.57	0.968	0.595	0.908	0.661
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.592	0.965	0.60	0.908	0.665
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.594	0.964	0.597	0.907	0.663

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

Table 4. Prediction performance of RF classifier on PreDNA dataset using different distance cut-offs.

Cut-Off	Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
3.5 Å	$f_{PSSM} + f_{OBV} + f_{SS}$	0.428	0.977	0.458	0.867	0.451
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.511	0.97	0.488	0.885	0.488
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.539	0.962	0.475	0.888	0.488
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.539	0.961	0.47	0.886	0.488
5.0 Å	$f_{PSSM} + f_{OBV} + f_{SS}$	0.395	0.98	0.488	0.858	0.530
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.48	0.968	0.511	0.874	0.563
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.506	0.962	0.51	0.873	0.560
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.499	0.96	0.498	0.871	0.555

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

2.2. Position Analysis of the MYC-MAX Protein

The proto-oncobenic transcription factor MYC-MAX (PDB-Entry 1NKP) is a heterodimer protein complex that is active in cell proliferation and is over-expressed in many different cancer types [38]. MYC-MAX transcription factors bind to Enhancer boxes (a core element of the promoter that consists of six nucleotides) and activate transcription of the underlying genes [39].

The amino acid chain of MYC protein consists of 88 residues, ten of which are known DNA-binding sites indicating that their distances to DNA are less than 3.5 Å. Applying RF classifier, which takes a majority vote among the random tree classifiers, with our first feature ($f_{\mathbb{JSD}}$) combined with existing features, we predicted in total 17 residue positions to be DNA-binding in MYC protein. Seven out of these positions (H906, N907, E910, R913, R914, P938, K939) correspond to the true DNA-binding sites of this protein. While the sites R913, R914, P938, and K939 could also be identified by RF classifier without using our new $\mathbb{JSD}$-based features, the remaining three binding sites could only be detected using our features (for details see Table 5 and Figure 1). Interestingly, using $f_{\mathbb{JSD}\text{-t}}$ together with f_{PSSM}, f_{OBV}, and f_{SS}, the RF classifier correctly predicted these seven positions again as binding sites.

The second protein in the proto-oncobenic transcription factor complex is the MAX protein which consists of 83 residues including nine DNA-binding sites. Using $f_{\mathbb{JSD}}$ or $f_{\mathbb{JSD}\text{-t}}$ together with existing features individually, we observed 14 and 13 residue positions to be DNA-binding in MAX protein, respectively. Eight of the predicted positions (H207, N208, E211, R212, R214, R215, S238, R239) found by using either of our both features are true DNA-binding sites in MAX protein. However, without using our new features the RF classifier could only identify two (S238, R239) out of nine true DNA-binding sites in MAX protein (for details see Table 5 and Figure 1). Further, we observed that, the usage of $f_{\mathbb{JSD}\text{-t}}$ leads to the reduction of false positive predictions in identifying DNA-binding sites in MAX protein.

Table 5. Prediction performance of RF classifier on different features using a cut-off of 3.5 Å for MYC-MAX protein complex (Protein Data Bank (PDB)-Entry 1NKP).

Protein	Feature	Sensitivity	Specificity	MCC
MYC	$f_{PSSM} + f_{OBV} + f_{SS}$	0.30	0.941	0.282
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.70	0.853	0.448
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.70	0.853	0.448
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.70	0.868	0.470
MAX	$f_{PSSM} + f_{OBV} + f_{SS}$	0.222	1.0	0.447
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.888	0.906	0.664
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.888	0.922	0.697
	$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.889	0.922	0.697

MCC: Matthews correlation coefficient.

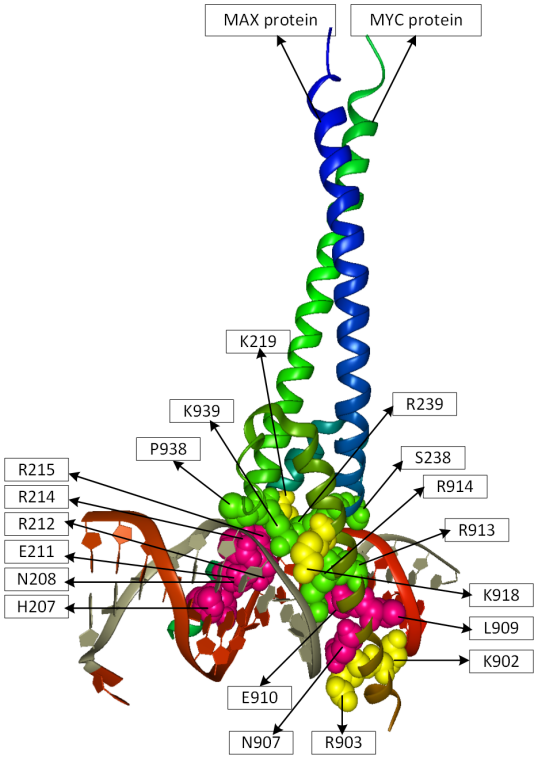

Figure 1. DNA-binding sites in proto-oncobenic transcription factor MYC-MAX protein complex (PDB-Entry 1NKP). Green spheres denote positions of the DNA-binding sites in both proteins which are detected by RF classifier either using the existing features (f_{PSSM}, f_{OBV}, and f_{SS}) alone or combining our new features with these existing features together. Purple spheres show the localization of additional binding sites which were only found by RF classifier using our new features with existing features. Moreover, there are further three binding sites in MYC protein and one binding site in MAX protein, shown with yellow spheres, that could not be identified by the classifier.

Moreover, when statistically evaluating both of our features, we observed that using our sequence-based features RF classifier reaches a significantly improved performance in identifying DNA-binding sites of both proteins with significantly higher sensitivity and MCC values whereas the specificity is moderately decreased. The simultaneous usage of both of our features together with f_{PSSM}, f_{OBV}, and f_{SS} could result in the decrement of specificity or MCC values. The details are presented in Table 5.

3. Materials and Methods

In this section, we describe in particular the data we have used and our new residue-wise features designed to predict DNA-binding sites in proteins.

3.1. Materials

To compile our data needed for training and test, we started with the DBP-374 data set of representative protein-DNA complexes from the Protein Data Bank (PDB) [40] published by Wu et al. [5]. Having performed a comparison with the new PDB version, we calculate for every remaining protein a multiple sequence alignment (MSA) using HHblits and the UniProt20 database (version from June 2015) [41]. We eliminated all proteins, the MSA of which has less than 125 rows, so that we finally ended up with a dataset of 263 protein-DNA complexes and associated MSAs. To obtain our results we perform a five-fold cross validation.

As in [5], an amino acid residue is regarded as a binding site, if it contains at least one atom at distance of less than or equal to 3.5 Å or 5 Å from any atom of DNA molecule in the DNA-protein complex. Otherwise it is treated as non-binding site. For the distance cut-off of 3.5 Å, our set contains 4298 binding sites and 44,805 non-binding sites. For the distance cut-off of 5 Å, however, our data set contains 7211 binding sites and 41,892 non-binding sites.

3.2. Methods

Let M be a multiple sequence alignment, where its first row represents the protein under study. Every residue of that protein is then uniquely determined by its column. In what follows, we identify the residues of the protein with their columns of the MSA.

Grosse et al. [35] pointed out that the Jensen–Shannon divergence ($\mathbb{JSD}$) is extremely useful when it comes to discriminate between two (or more) sources. Capra and Singh [34] carefully discussed several information theoretic measures like Shannon entropy, von Neumann entropy, relative entropy, and sum-of-pair measures to assess sequence conservation. They were the first using $\mathbb{JSD}$ in this context and stated its superiority. Gültas et al. [32] showed that the Jensen–Shannon divergence in the context of quantum information theory is of remarkable power. These three articles encouraged us to use $\mathbb{JSD}$ in this study. Our first idea is to design a new feature for the prediction of DNA-binding sites in proteins which leverages the *Jensen–Shannon divergence*

$$\mathbb{JSD}\left(\mathbf{p}_k \,\|\mathbf{p}_{nd}\right) := \mathbb{H}\left(\left(\mathbf{p}_k + \mathbf{p}_{nd}\right)/2\right) - \left(\mathbb{H}\left(\mathbf{p}_k\right) + \mathbb{H}\left(\mathbf{p}_{nd}\right)\right)/2. \tag{1}$$

Therein, $\mathbf{p}_k$ is the empirical amino acid distribution of the k-th column of the query MSA M, and $\mathbf{p}_{nd}$ is the *null distribution* taken over all non-binding sites of our training data.

More precisely, we represent every column k of every MSA M considered by a 20×20 counting matrix $C\left(M_{\bullet k}\right)$. The matrix C is symmetric and its rows as well as columns are indexed by the 20 amino acids. For every ordered pair of amino acids $\left(a, a'\right)$, the matrix coefficient $C\left(M_{\bullet k}\right)_{aa'}$ is equal to the number of ordered pairs (i, j) $(i \neq j)$ of row indices of M such that $M_{ik} = a$ and $M_{jk} = a'$.

To compute the null distribution $\mathbf{p}_{nd}$, we first set up the 20×20 counting matrix $\mathcal{C}_{nd}$ using our training data. $\mathcal{C}_{nd}$ is the sum over all matrices $C\left(M_{\bullet k}\right)$, where M ranges over all training MSAs and k ranges over all non-binding site columns of M. Next, the rows of $\mathcal{C}_{nd}$ are added up. Finally, the resulting row vector is normalized to obtain $\mathbf{p}_{nd}$.

There is nothing wrong with the idea that a large value $\mathbb{JSD}\left(\mathbf{p}_k \,\|\mathbf{p}_{nd}\right)$ indicates that k is a DNA-binding residue. However, no information on binding sites is integrated. Only the non-binding sites of our training data are used to compute $\mathbf{p}_{nd}$. As we have seen in [32] and [36], transforming empirical amino acid distributions of MSA columns by a carefully designed doubly stochastic matrix is an effective way to integrate the binding site signals. To this end, we first set up a counting matrix $\mathcal{C}_{bind}$ in a way similar to that of calculating the matrix $\mathcal{C}_{nd}$. The difference is that the variable column index k now ranges over all binding site columns of the training MSAs. Taking the counting matrix $\mathcal{C}_{bind}$ as input, the doubly stochastic matrix $\mathcal{D}$ is computed by means of the canonical row-column normalization procedure [42].

Let M be the query MSA having ℓ columns. Compared with [32] and [36], we enhance the effect of transforming M's empirical column distributions by means of the doubly stochastic matrix $\mathcal{D}$ just defined. Let k be a column index of M. First, we compute the matrix product $C^{(t)}\left(M_{\bullet k}\right) := C\left(M_{\bullet k}\right) \cdot \mathcal{D}$. Second, we add up all of $C^{(t)}\left(M_{\bullet k}\right)$'s rows. Finally, we normalize the resulting row to obtain the transformed empirical row distribution $\mathbf{p}_k^{(t)}$.

We define two *window scores* $\text{score}_{\mathbb{JSD},M}\left(k\right)$ and $\text{score}_{\mathbb{JSD}\text{-t},M}\left(k\right)$ of residue k w.r.t. query MSA M, where the window $\mathfrak{w}(k)$ surrounding k formally equals $\{k-3, k-2, k-1, k, k+1, k+2, k+3\} \cap \{1, 2, \ldots, \ell\}$. Clearly, if $k \in \{4, 5, \ldots, \ell-3\}$, $|\mathfrak{w}(k)| = 7$. Otherwise $|\mathfrak{w}(k)| \in \{4, 5, 6\}$. Recapitulate that for any real x the binomial coefficient $\binom{x}{2}$ equals $x(x-1)/2$. We define the scores as follows.

$$\text{score}_{\mathbb{JSD},M}(k) := \frac{\sum_{\iota \in \mathfrak{w}(k)} (4 - |k - \iota|)\, \mathbb{JSD}(\mathbf{p}_{k+\iota} \,\|\, \mathbf{p}_{nd})}{16 - \binom{8 - |\mathfrak{w}(k)|}{2}} \tag{2}$$

$$\text{score}_{\mathbb{JSD}\text{-t},M}(k) := \frac{\sum_{\iota \in \mathfrak{w}(k)} (4 - |k - \iota|)\, \mathbb{JSD}\left(\mathbf{p}^{(t)}_{k+\iota} \,\|\, \mathbf{p}_{nd}\right)}{16 - \binom{8 - |\mathfrak{w}(k)|}{2}} \tag{3}$$

The preceding two score definitions are motivated as follows. Bartlett et al. [43] and Panchenko et al. [44] pointed out that exploiting conservation properties of spatial neighbors is useful to predict a residue as functionally important. Since the 3D structures are often unavailable, Capra and Singh [34] developed a window score for such predictions. The concrete shape of our scores takes pattern form Janda et al. [45], who in turn refer to Fischer et al. [33]. Our scores are convex combinations of the Jensen–Shannon terms associated with the residues belonging to the surrounding window $\mathfrak{w}(k)$. The weights fall linearly in the distance from k.

In a last step, we transform two window scores according to Equations (2) and (3) with respect to the query MSA M into final scores using the Equations (4) and (5), respectively. To this end, for every column index $k \in \{1, 2, \ldots, \ell\}$ of M we define:

$$\mathrm{f}_{\mathbb{JSD},M}(k) := \frac{\left|\{k' \mid 1 \le k' \le \ell,\ \text{score}_{\mathbb{JSD},M}(k) \ge \text{score}_{\mathbb{JSD},M}(k')\}\right|}{\ell} \tag{4}$$

$$\mathrm{f}_{\mathbb{JSD}\text{-t},M}(k) := \frac{\left|\{k' \mid 1 \le k' \le \ell,\ \text{score}_{\mathbb{JSD}\text{-t},M}(k) \ge \text{score}_{\mathbb{JSD}\text{-t},M}(k')\}\right|}{\ell}. \tag{5}$$

The Equations (4) and (5) are basically the determination of the percentage of scores below the current one at index k. This transformation procedure is essential because it converts MSA-dependent window scores to MSA-independent scores.

To demonstrate the benefit of our new features, we adopt the features f_{PSSM}, f_{OBV} and f_{SS} devised in [5]. Together with our two new features $\mathrm{f}_{\mathbb{JSD}}$ and $\mathrm{f}_{\mathbb{JSD}\text{-t}}$, we plugged them into the Random Forest (RF) classifier [46] (see Tables 1 and 2 for the combinations we used). For the RF implementation we used the WEKA data mining software [47].

To deal with the imbalanced data problem, we applied bagging techniques suggested in [48]. Since we make use of five-fold cross validation, we randomly split the dataset into 5 roughly equal-sized parts. Every training phase performed on 4 parts consists of 11 sub-phases. In each such sub-phase we randomly draw twice as many non-binding sites as there are binding sites. We then construct a Random Forest (RF) taking those non-binding sites and all binding sites of the 4 parts as input. Finally, for each instance of the validation part the majority vote of above 11 RF classifiers was taken.

4. Discussion

Our results show that combining either feature $\mathrm{f}_{\mathbb{JSD}\text{-t}}$ or feature $\mathrm{f}_{\mathbb{JSD}}$ with the three features f_{PSSM}, f_{OBV} and f_{SS} we have adopted from [5] clearly boosts the performance of the RF-based classifier in identifying the DNA-binding sites in proteins, where feature $\mathrm{f}_{\mathbb{JSD}\text{-t}}$ generally reaches a slightly better performance than feature $\mathrm{f}_{\mathbb{JSD}}$.

Although our two new features and PSSMs are derived from MSAs, Tables 1 and 2 clearly demonstrate that these approaches carry distinct information. Thus they capture different kinds of evolutionary information. The reason for this essential difference can be explained based on the underlying algorithms. While the PSSM approach consists of statistic which indicates how likely a certain amino acids occurs at a certain position, our $\mathbb{JSD}$-based approach measures the divergence of a certain distribution to a known non-binding site distribution.

The superiority of feature $f_{\mathbb{JSD}\text{-t}}$ to feature $f_{\mathbb{JSD}}$ deserves an explanation attempt. Feature $f_{\mathbb{JSD}}$ does not integrate any information on DNA-binding sites. Only training non-binding sites are used. In contrast, feature $f_{\mathbb{JSD}\text{-t}}$ additionally uses a doubly stochastic matrix gained from the training binding sites. The effect on empirical amino acid column distributions of the transformation we have devised using that matrix is the following. The empirical column probabilities of amino acids are merged, if it is very likely to co-observe them in a binding site column. Since the amino acid content of binding site columns and non-binding site columns differ, the distance between $f_{\mathbb{JSD}\text{-t},M}(k)$ and $f_{\mathbb{JSD}\text{-t},M}(k')$ is larger and more significant than the distance between $f_{\mathbb{JSD},M}(k)$ and $f_{\mathbb{JSD},M}(k')$, where k is a binding site column of MSA M, and k' is a non-binding site column.

At first glance it is surprising that adding both feature $f_{\mathbb{JSD}\text{-t}}$ and feature $f_{\mathbb{JSD}}$ to the feature triplet $(f_{PSSM}, f_{OBV}, f_{SS})$ is worse than adding feature $f_{\mathbb{JSD}\text{-t}}$ alone. Taking into account what we have mentioned in the preceding paragraph, it turns out that if feature $f_{\mathbb{JSD}\text{-t}}$ is already there, feature $f_{\mathbb{JSD}}$ may increase the noise.

5. Conclusions

In this work, we report a new sequence-based feature extraction method for the identification of DNA binding sites in proteins. For this purpose, we adopt the ideas from Capra et al. [34] and our previous studies CMF [36] and QCMF [32]. Our approach is an information theoretic method that applies the Jensen–Shannon divergence ($\mathbb{JSD}$) for amino acid distributions of each site in a protein in two different ways. First, the $\mathbb{JSD}$ is applied to quantify the differences between observed amino acid distributions of sites and the background distribution of non-binding sites. Second, we transform the observed distributions of sites through a doubly stochastic matrix to incorporate biochemical signals of binding residues in the calculation of $\mathbb{JSD}$ that results in the intensification of the DNA-binding residue signals from the non-binding signals. The results of our study show that the additional usage of our new features ($f_{\mathbb{JSD}\text{-t}}$ or feature $f_{\mathbb{JSD}}$) in combination with existing features is significantly boosts the performance of RF classifier in identifying DNA binding sites in proteins. Our results further indicate the importance of our second feature ($f_{\mathbb{JSD}\text{-t}}$) since taking into account the binding site signals in the calculation of $\mathbb{JSD}$ metric, the characteristics of DNA binding residues are enhanced. As a consequence, an intensification of the signal caused by DNA binding sites from non-binding sites occurs and thus the classifier achieves its improved performance.

Acknowledgments: We thank our colleagues Edgar Wingender, Martin Haubrock and Sebastian Zeidler for their helpful advice and insights at early stages of this project. We acknowledge support by the German Research Foundation and the Open Access Publication Funds of the Göttingen University.

Author Contributions: Mehmet Gültas developed the model. Stephan Waack adjusted the model together with Mehmet Gültas. Truong Khanh Linh Dang developed the model together with Mehmet Gültas, designed and implemented the tool and interpreted the results together with Cornelia Meckbach, Rebecca Tacke and Mehmet Gültas. Cornelia Meckbach and Rebecca Tacke studied the DNA binding sites in MYC-MAX protein complex. Mehmet Gültas conceived of and managed the project and wrote the final version of the manuscript. All authors have read and approved the final manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

The detailed performance of the RF classifier on different features using different cut-offs for RBscore and PreDNA datasets.

Appendix A.1. Performance Measures with Standard Error

Table A1. Prediction performance of Random Forest (RF) classifier on different features using a cut-off of 3.5 Å. The prediction system was evaluated by five-fold cross validation.

Feature	Sensitivity ± SE(%)	Specificity ± SE(%)	MCC ± SE(%)
f_{PSSM}	29.2 ± 2.20	96.3 ± 0.46	30.7 ± 0.95
f_{PSSM} + $f_{\mathbb{JSD}}$	38.5 ± 3.04	94.9 ± 0.57	34.9 ± 1.7
f_{PSSM} + $f_{\mathbb{JSD}\text{-t}}$	41.0 ± 3.23	93.9 ± 0.57	35.0 ± 1.85
f_{PSSM} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	41.4 ± 3.42	94.0 ± 0.51	34.8 ± 2.07
f_{PSSM} + f_{SS}	33.9 ± 2.32	95.8 ± 0.37	33.4 ± 1.36
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}}$	41.6 ± 3.05	95.0 ± 0.46	37.8 ± 2.19
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}\text{-t}}$	44.1 ± 3.12	94.0 ± 0.43	37.2 ± 2.37
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	43.9 ± 3.14	94.0 ± 0.40	37.0 ± 2.25
f_{PSSM} + f_{OBV} + f_{SS}	36.7 ± 2.07	96.8 ± 0.27	39.8 ± 1.58
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}}$	42.2 ± 2.70	95.8 ± 0.42	40.9 ± 1.95
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}\text{-t}}$	44.7 ± 3.05	95.0 ± 0.38	40.3 ± 1.98
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	44.4 ± 3.12	94.7 ± 0.39	39.3 ± 2.02

Table A2. Prediction performance of Random Forest (RF) classifier on different features using a cut-off of 5.0 Å. The prediction system was evaluated by five-folds cross validation.

Feature	Sensitivity ± SE(%)	Specificity ± SE(%)	MCC ± SE(%)
f_{PSSM}	28.6 ± 2.56	96.6 ± 0.47	35.0 ± 1.43 5
f_{PSSM} + $f_{\mathbb{JSD}}$	39.5 ± 2.89	95.0 ± 0.55	40.7 ± 1.99
f_{PSSM} + $f_{\mathbb{JSD}\text{-t}}$	41.8 ± 3.02	94.3 ± 0.62	41.1 ± 2.05
f_{PSSM} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	42.6 ± 3.25	94.2 ± 0.54	41.4 ± 2.37
f_{PSSM} + f_{SS}	33.4 ± 2.34	96.3 ± 0.38	38.6 ± 1.90
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}}$	42.4 ± 2.97	95.1 ± 0.61	43.6 ± 2.43
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}\text{-t}}$	44.8 ± 2.99	94.4 ± 0.56	43.8 ± 2.45
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	44.5 ± 3.04	94.4 ± 0.50	43.4 ± 2.35
f_{PSSM} + f_{OBV} + f_{SS}	33.7 ± 2.48	97.5 ± 0.35	43.1 ± 2.05
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}}$	41.9 ± 2.89	95.8 ± 0.55	45.0 ± 2.39
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}\text{-t}}$	43.9 ± 2.89	95.2 ± 0.48	45.3 ± 2.32
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	44.2 ± 2.91	94.9 ± 0.54	44.5 ± 2.24

Appendix A.2. RBscore Dataset Analysis

Table A3. The detailed prediction performance of Random Forest (RF) classifier on different features using a cut-off of 3.5 Å.

Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
f_{PSSM}	0.458	0.974	0.476	0.866	0.460
f_{PSSM} + $f_{\mathbb{JSD}}$	0.56	0.965	0.514	0.894	0.518
f_{PSSM} + $f_{\mathbb{JSD}\text{-t}}$	0.597	0.957	0.511	0.899	0.523
f_{PSSM} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	0.591	0.958	0.511	0.90	0.526
f_{PSSM} + f_{SS}	0.512	0.97	0.501	0.878	0.476
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}}$	0.581	0.96	0.511	0.899	0.520
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}\text{-t}}$	0.611	0.953	0.508	0.903	0.526
f_{PSSM} + f_{SS} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	0.613	0.953	0.509	0.902	0.528
f_{PSSM} + f_{OBV} + f_{SS}	0.517	0.976	0.534	0.896	0.528
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}}$	0.58	0.967	0.54	0.907	0.543
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}\text{-t}}$	0.612	0.963	0.546	0.910	0.551
f_{PSSM} + f_{OBV} + f_{SS} + $f_{\mathbb{JSD}}$ + $f_{\mathbb{JSD}\text{-t}}$	0.601	0.962	0.531	0.909	0.546

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

Table A4. The detailed prediction performance of Random Forest (RF) classifier on different features using a cut-off of 5.0 Å.

Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
f_{PSSM}	0.445	0.977	0.528	0.873	0.589
$f_{PSSM} + f_{\mathbb{JSD}}$	0.553	0.968	0.579	0.899	0.643
$f_{PSSM} + f_{\mathbb{JSD}\text{-t}}$	0.57	0.962	0.572	0.900	0.642
$f_{PSSM} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.569	0.963	0.574	0.895	0.642
$f_{PSSM} + f_{SS}$	0.49	0.973	0.547	0.880	0.602
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}}$	0.578	0.963	0.583	0.902	0.648
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.605	0.958	0.587	0.904	0.652
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.603	0.959	0.587	0.902	0.653
$f_{PSSM} + f_{OBV} + f_{SS}$	0.499	0.98	0.584	0.895	0.641
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.57	0.968	0.595	0.908	0.661
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.592	0.965	0.60	0.908	0.665
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.594	0.964	0.597	0.907	0.663

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

Appendix A.3. PreDNA Dataset Analysis

Table A5. The detailed prediction performance of Random Forest (RF) classifier on different features using a cut-off of 3.5 Å.

Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
f_{PSSM}	0.378	0.977	0.41	0.840	0.391
$f_{PSSM} + f_{\mathbb{JSD}}$	0.498	0.963	0.448	0.865	0.453
$f_{PSSM} + f_{\mathbb{JSD}\text{-t}}$	0.543	0.953	0.445	0.869	0.451
$f_{PSSM} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.538	0.956	0.453	0.869	0.455
$f_{PSSM} + f_{SS}$	0.393	0.975	0.417	0.847	0.402
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}}$	0.501	0.966	0.461	0.872	0.463
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.545	0.959	0.465	0.876	0.468
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.523	0.958	0.449	0.875	0.465
$f_{PSSM} + f_{OBV} + f_{SS}$	0.428	0.977	0.458	0.867	0.451
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.511	0.97	0.488	0.885	0.488
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.539	0.962	0.475	0.888	0.488
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.539	0.961	0.47	0.886	0.488

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

Table A6. The detailed prediction performance of Random Forest (RF) classifier on different features using a cut-off of 5.0 Å.

Feature	Sensitivity	Specificity	MCC	AUC-ROC	AUC-PR
f_{PSSM}	0.373	0.979	0.463	0.833	0.496
$f_{PSSM} + f_{\mathbb{JSD}}$	0.485	0.962	0.495	0.858	0.540
$f_{PSSM} + f_{\mathbb{JSD}\text{-t}}$	0.496	0.953	0.475	0.858	0.534
$f_{PSSM} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.495	0.955	0.479	0.857	0.535
$f_{PSSM} + f_{SS}$	0.389	0.977	0.47	0.839	0.501
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}}$	0.49	0.963	0.501	0.863	0.550
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.503	0.957	0.492	0.865	0.547
$f_{PSSM} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.504	0.958	0.497	0.865	0.550
$f_{PSSM} + f_{OBV} + f_{SS}$	0.395	0.98	0.488	0.858	0.530
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}}$	0.48	0.968	0.511	0.874	0.563
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}\text{-t}}$	0.506	0.962	0.51	0.873	0.560
$f_{PSSM} + f_{OBV} + f_{SS} + f_{\mathbb{JSD}} + f_{\mathbb{JSD}\text{-t}}$	0.499	0.96	0.498	0.871	0.555

MCC: Matthews correlation coefficient; AUC-ROC: area under the receiver operating characteristics (ROC) curve; AUC-PR: area under the precision-recall curve.

References

1. Liu, B.; Wang, S.; Wang, X. DNA binding protein identification by combining pseudo amino acid composition and profile-based protein representation. *Sci. Rep.* **2015**, *5*, 15479.
2. Miao, Z.; Westhof, E. Prediction of nucleic acid binding probability in proteins: A neighboring residue network based score. *Nucleic Acids Res.* **2015**, *43*, 5340–5351.
3. Si, J.; Zhang, Z.; Lin, B.; Schroeder, M.; Huang, B. MetaDBSite: A meta approach to improve protein DNA-binding sites prediction. *BMC Syst. Biol.* **2011**, *5* (Suppl. S1), S7.
4. Ma, X.; Guo, J.; Liu, H.D.; Xie, J.M.; Sun, X. Sequence-based prediction of DNA-binding residues in proteins with conservation and correlation information. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2012**, *9*, 1766–1775.
5. Wu, J.; Liu, H.; Duan, X.; Ding, Y.; Wu, H.; Bai, Y.; Sun, X. Prediction of DNA-binding residues in proteins from amino acid sequences using a random forest model with a hybrid feature. *Bioinformatics* **2009**, *25*, 30–35.
6. Liu, B.; Xu, J.; Fan, S.; Xu, R.; Zhou, J.; Wang, X. PseDNA-Pro: DNA-Binding Protein Identification by Combining Chou's PseAAC and Physicochemical Distance Transformation. *Mol. Inform.* **2015**, *34*, 8–17.
7. Xu, R.; Zhou, J.; Wang, H.; He, Y.; Wang, X.; Liu, B. Identifying DNA-binding proteins by combining support vector machine and PSSM distance transformation. *BMC Syst. Biol.* **2015**, *9* (Suppl. S1), S10.
8. Dong, Q.; Wang, S.; Wang, K.; Liu, X.; Liu, B. Identification of DNA-binding proteins by auto-cross covariance transformation. In Proceedings of the 2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Washington, DC, USA, 9–12 November 2015; pp. 470–475.
9. Wei, L.; Tang, J.; Zou, Q. Local-DPP: An improved DNA-binding protein prediction method by exploring local evolutionary information. *Inf. Sci.* **2016**, in press.
10. Waris, M.; Ahmad, K.; Kabir, M.; Hayat, M. Identification of DNA binding proteins using evolutionary profiles position specific scoring matrix. *Neurocomputing* **2016**, *199*, 154–162.
11. Zhou, J.; Xu, R.; He, Y.; Lu, Q.; Wang, H.; Kong, B. PDNAsite: Identification of DNA-binding Site from Protein Sequence by Incorporating Spatial and Sequence Context. *Sci. Rep.* **2016**, *6*, 27653.
12. Jones, S.; Shanahan, H.P.; Berman, H.M.; Thornton, J.M. Using electrostatic potentials to predict DNA-binding sites on DNA-binding proteins. *Nucleic Acids Res.* **2003**, *31*, 7189–7198.
13. Ahmad, S.; Gromiha, M.M.; Sarai, A. Analysis and prediction of DNA-binding proteins and their binding residues based on composition, sequence and structural information. *Bioinformatics* **2004**, *20*, 477–486.
14. Bhardwaj, N.; Langlois, R.E.; Zhao, G.; Lu, H. Structure based prediction of binding residues on DNA-binding proteins. In Proceedings of the IEEE 27th Annual International Conference of the Engineering in Medicine and Biology Society (IEEE-EMBS 2005), Shanghai, China, 1–4 September 2005; pp. 2611–2614.
15. Ahmad, S.; Sarai, A. PSSM-based prediction of DNA binding sites in proteins. *BMC Bioinform.* **2005**, *6*, 33.
16. Kuznetsov, I.B.; Gou, Z.; Li, R.; Hwang, S. Using evolutionary and structural information to predict DNA-binding sites on DNA-binding proteins. *Proteins* **2006**, *64*, 19–27.
17. Wang, L.; Brown, S.J. Prediction of DNA-binding residues from sequence features. *J. Bioinform. Comput. Biol.* **2006**, *4*, 1141–1158.
18. Wang, L.; Brown, S.J. BindN: A web-based tool for efficient prediction of DNA and RNA binding sites in amino acid sequences. *Nucleic Acids Res.* **2006**, *34*, W243–W248.
19. Ofran, Y.; Mysore, V.; Rost, B. Prediction of DNA-binding residues from sequence. *Bioinformatics* **2007**, *23*, i347–i353.
20. Siggers, T.W.; Honig, B. Structure-based prediction of C2H2 zinc-finger binding specificity: Sensitivity to docking geometry. *Nucleic Acids Res.* **2007**, *35*, 1085–1097.
21. Tjong, H.; Zhou, H.X. DISPLAR: An accurate method for predicting DNA-binding sites on protein surfaces. *Nucleic Acids Res.* **2007**, *35*, 1465–1477.
22. Nimrod, G.; Schushan, M.; Szilágyi, A.; Leslie, C.; Ben-Tal, N. iDBPs: A web server for the identification of DNA binding proteins. *Bioinformatics* **2010**, *26*, 692–693.
23. Wang, L.; Huang, C.; Yang, M.Q.; Yang, J.Y. BindN+ for accurate prediction of DNA and RNA-binding residues from protein sequence features. *BMC Syst. Biol.* **2010**, *4* (Suppl. S1), S3.
24. Miao, Z.; Westhof, E. A Large-Scale Assessment of Nucleic Acids Binding Site Prediction Programs. *PLoS Comput. Biol.* **2015**, *11*, e1004639.
25. Yan, J.; Friedrich, S.; Kurgan, L. A comprehensive comparative review of sequence-based predictors of DNA-and RNA-binding residues. *Brief. Bioinform.* **2015**, *17*, 88–105.

26. Yan, C.; Terribilini, M.; Wu, F.; Jernigan, R.L.; Dobbs, D.; Honavar, V. Predicting DNA-binding sites of proteins from amino acid sequence. *BMC Bioinform.* **2006**, *7*, 262.
27. Hwang, S.; Gou, Z.; Kuznetsov, I.B. DP-Bind: A web server for sequence-based prediction of DNA-binding residues in DNA-binding proteins. *Bioinformatics* **2007**, *23*, 634–636.
28. Huang, Y.F.; Huang, C.C.; Liu, Y.C.; Oyang, Y.J.; Huang, C.K. DNA-binding residues and binding mode prediction with binding-mechanism concerned models. *BMC Genom.* **2009**, *10* (Suppl. S3), S23.
29. Wong, K.C.; Li, Y.; Peng, C.; Moses, A.M.; Zhang, Z. Computational learning on specificity-determining residue-nucleotide interactions. *Nucleic Acids Res.* **2015**, *43*, 10180–10189.
30. Wang, L.; Yang, M.Q.; Yang, J.Y. Prediction of DNA-binding residues from protein sequence information using random forests. *BMC Genom.* **2009**, *10* (Suppl. S1), S1.
31. Eggeling, R.; Roos, T.; Myllymäki, P.; Grosse, I. Inferring intra-motif dependencies of DNA binding sites from ChIP-seq data. *BMC Bioinform.* **2015**, *16*, doi:10.1186/s12859-015-0797-4.
32. Gültas, M.; Düzgün, G.; Herzog, S.; Jäger, S.J.; Meckbach, C.; Wingender, E.; Waack, S. Quantum coupled mutation finder: Predicting functionally or structurally important sites in proteins using quantum Jensen–Shannon divergence and CUDA programming. *BMC Bioinform.* **2014**, *15*, 96.
33. Fischer, J.; Mayer, C.E.; Söding, J. Prediction of protein functional residues from sequence by probability density estimation. *Bioinformatics* **2008**, *24*, 613–620.
34. Capra, J.A.; Singh, M. Predicting functionally important residues from sequence conservation. *Bioinformatics* **2007**, *23*, 1875–1882.
35. Grosse, I.; Bernaola-Galván, P.; Carpena, P.; Román-Roldán, R.; Oliver, J.; Stanley, H.E. Analysis of symbolic sequences using the Jensen–Shannon divergence. *Phys. Rev. E* **2002**, *65*, 041905.
36. Gültas, M.; Haubrock, M.; Tüysüz, N.; Waack, S. Coupled mutation finder: A new entropy-based method quantifying phylogenetic noise for the detection of compensatory mutations. *BMC Bioinform.* **2012**, *13*, 225.
37. Li, T.; Li, Q.Z.; Liu, S.; Fan, G.L.; Zuo, Y.C.; Peng, Y. PreDNA: Accurate prediction of DNA-binding sites in proteins by integrating sequence and geometric structure information. *Bioinformatics* **2013**, *29*, 678–685,
38. Krall, A.; Brunn, J.; Kankanala, S.; Peters, M.H. A simple contact mapping algorithm for identifying potential peptide mimetics in protein–protein interaction partners. *Proteins* **2014**, *82*, 2253–2262.
39. Nair, S.K.; Burley, S.K. X-ray structures of Myc-Max and Mad-Max recognizing DNA: Molecular bases of regulation by proto-oncogenic transcription factors. *Cell* **2003**, *112*, 193–205.
40. Berman, H.M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T.N.; Weissig, H.; Shindyalov, I.N.; Bourne, P.E. The protein data bank. *Nucleic Acids Res.* **2000**, *28*, 235–242.
41. Remmert, M.; Biegert, A.; Hauser, A.; Söding, J. HHblits: Lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nat. Methods* **2012**, *9*, 173–175.
42. Cappellini, V.; Sommer, H.J.; Bruzda, W.; Zyczkowski, K. Random bistochastic matrices. *J. Phys. A Math. Theor.* **2009**, *42*, 36.
43. Bartlett, G.J.; Porter, C.T.; Borkakoti, N.; Thornton, J.M. Analysis of catalytic residues in enzyme active sites. *J. Mol. Biol.* **2002**, *324*, 105–121.
44. Panchenko, A.R.; Kondrashov, F.; Bryant, S. Prediction of functional sites by analysis of sequence and structure conservation. *Protein Sci.* **2004**, *13*, 884–892.
45. Janda, J.O.; Busch, M.; Kück, F.; Porfenenko, M.; Merkl, R. CLIPS-1D: Analysis of multiple sequence alignments to deduce for residue-positions a role in catalysis, ligand-binding, or protein structure. *BMC Bioinform.* **2012**, *13*, 55.
46. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32.
47. Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I.H. The WEKA data mining software: An update. *ACM SIGKDD Explor. Newsl.* **2009**, *11*, 10–18.
48. Breiman, L. Bagging predictors. *Mach. Learn.* **1996**, *24*, 123–140.

© 2016 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC-BY) license (http://creativecommons.org/licenses/by/4.0/).

A.3. CRF-based Models for Protein-protein Interaction Site Predictions

Dong *et al. BMC Bioinformatics* 2014, **15**:277
http://www.biomedcentral.com/1471-2105/15/277

BMC Bioinformatics

METHODOLOGY ARTICLE Open Access

CRF-based models of protein surfaces improve protein-protein interaction site predictions

Zhijie Dong[1], Keyu Wang[1], Truong Khanh Linh Dang[1], Mehmet Gültas[2], Marlon Welter[1], Torsten Wierschin[3], Mario Stanke[3] and Stephan Waack[1*]

Abstract

Background: The identification of protein-protein interaction sites is a computationally challenging task and important for understanding the biology of protein complexes. There is a rich literature in this field. A broad class of approaches assign to each candidate residue a real-valued score that measures how likely it is that the residue belongs to the interface. The prediction is obtained by thresholding this score.
Some probabilistic models classify the residues on the basis of the posterior probabilities. In this paper, we introduce pairwise conditional random fields (pCRFs) in which edges are not restricted to the backbone as in the case of linear-chain CRFs utilized by Li *et al.* (2007). In fact, any 3D-neighborhood relation can be modeled. On grounds of a generalized Viterbi inference algorithm and a piecewise training process for pCRFs, we demonstrate how to utilize pCRFs to enhance a given residue-wise score-based protein-protein interface predictor on the surface of the protein under study. The features of the pCRF are solely based on the interface predictions scores of the predictor the performance of which shall be improved.

Results: We performed three sets of experiments with synthetic scores assigned to the surface residues of proteins taken from the data set *PlaneDimers* compiled by Zellner *et al.* (2011), from the list published by Keskin *et al.* (2004) and from the very recent data set due to Cukuroglu *et al.* (2014). That way we demonstrated that our pCRF-based enhancer is effective given the interface residue score distribution and the non-interface residue score are unimodal. Moreover, the pCRF-based enhancer is also successfully applicable, if the distributions are only unimodal over a certain sub-domain. The improvement is then restricted to that domain. Thus we were able to improve the prediction of the *PresCont* server devised by Zellner *et al.* (2011) on *PlaneDimers*.

Conclusions: Our results strongly suggest that pCRFs form a methodological framework to improve residue-wise score-based protein-protein interface predictors given the scores are appropriately distributed. A prototypical implementation of our method is accessible at http://ppicrf.informatik.uni-goettingen.de/index.html.

Background

Protein-protein interactions are constitutive of almost every biological process. The ability to identify the residues that form the interaction sites of these complexes is necessary to understand them. In particular, it is the basis for new therapeutic approaches to treat diseases [1,2].

A great deal of work has been done on developing in-silico prediction methods. As already observed by Zhou *et al.* [3], these methods can be subdivided with respect to the kind of mathematical foundation invoked and with respect to the features or characteristics of the protein used.

Residue-wise score-based prediction methods

Let x_r be the data relevant for a residue r in a given protein chain. These methods then employ a function $f(x_r, \lambda)$, where λ are some coefficients which have been learned through the training. The value of $f(x_r, \lambda)$ then determines, whether r is rated as an interface or not. The linear regression method [4,5], the scoring function method [6-11], the neural network method [12-17], and the support vector machine method [18-25] are of this kind.

*Correspondence: waack@informatik.uni-goettingen.de
[1] Institute of Computer Science, University of Göttingen, Goldschmidtstr. 7, 37077 Göttingen, Germany
Full list of author information is available at the end of the article

BioMed Central

© 2014 Dong et al.; licensee BioMed Central Ltd. This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/2.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly credited. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated.

Probabilistic methods

Let $\mathbf{X}$ be the data relevant for a protein chain, where these data are assumed to stem from a random source thus obeying a random distribution. $\mathbf{X}$, which alternatively is called the observation, typically includes the structure. The label sequence of the residues $\mathbf{Y}$ that classifies each individual residue either as interface or as non-interface is assumed to be random, too. Typically, probabilistic methods use the conditional probability distribution $\mathbb{P}(\mathbf{Y} \mid \mathbf{X})$ to determine a classification $\mathbf{y}^*$ of the residues of maximal posterior probability $\mathbb{P}(\mathbf{y}^* \mid \mathbf{x})$. Naive Bayesian methods [26], Bayesian network methods [27], hidden Markov models (HMMs) [26], and linear-chain Conditional Random Fields taking the backbone as underlying graphical structure [28] fall in this category. Using posterior decoding on the basis of the forward-backward algorithm, both HMMs and CRFs are residue-wise score-based prediction methods, where the binary decision is made by thresholding the posterior probabilities of classifying the residues as interface.

Notations

We use Latin uppercase letters when referring to random aspects of the objects denoted by them. In contrast, lowercase letters denote arbitrarily chosen but fixed objects. In this context boldface letters indicate vectors, the corresponding non-boldface letters their coefficients.

The vast majority of methods use the 3D structure of the target protein chain in form of a PDB file as input [4-13,15,17-21,23-25]. However, a few methods are not requiring a 3D structure and rather use sequences only [14,16,22]. We here consider the problem with a given 3D structure of the target protein chain. Sequence-based input may include a multiple sequence alignment of related proteins from which, for example, sequence conservation can be inferred. When the 3D structure of an unbound binding partner is also available, protein-protein docking methods can be applied. This has also been exploited to provide feedback from docking to the more specific problem of interface prediction [29]. We here consider the case where the binding partner's 3D structure is not given. Nor requires the presented method the sequence of the binding partner. Albeit, we tested on homodimers only as we here rather focus on our new method rather than on features or types of proteins. The protein features used for interface prediction in the literature are reviewed in the Methods section as far as we make use of them in this article.

Most of the current studies for predicting interaction sites of proteins that use a probabilistic method are restricted by treating the residues of the proteins as independent vertices. Li *et al.* have taken the backbone neighborhood into account thus modeling the protein as a sequence [28] using what can be called a *line CRF* or linear-chain CRF. The features they define on the label pair of two backbone neighbors have the effect of smoothing the predicted labels along the protein sequence. Decisive is, however, that they were the first who used conditional random fields (CRFs) for interface prediction. CRFs in turn have come into use for solving sequence labeling problems due to Lafferty *et al.* [30]. See [31] for an overview. From the mathematical point of view they take advantage of the fact that they model the conditional probability $\mathbb{P}(\mathbf{Y} \mid \mathbf{X})$ rather than the joint probability $\mathbb{P}(\mathbf{Y}, \mathbf{X})$. Recently there has been an explosion of interest in conditional random fields (CRFs) with successful applications. It has been shown that CRFs have the abilities for solving sequence labeling problems like part-of-speech tagging (POST) [32] and natural language processing [33]. Furthermore in the web extraction problem, in which the web-sites are modeled as two dimensional grid graphs, CRFs perform well [34]. One of their outstanding benefits over many other statistical models is that a CRF can easily describe the dependencies of observations.

As proteins are folded into three dimensional structures, spatial relationships create dependencies between residues. For example, we find on the test data described below that the correlation coefficient between spatial neighbors that are not also sequence neighbors (distance ≤ 3.5 Å) is 0.45. This is only slightly lower than the correlation coefficient between residues that are sequence neighbors (0.49). As there are more than three times as many spatial pairs of neighbors than sequence neighbors at this threshold it is reasonable from a modeling standpoint to use a model that respects *all* dependencies induced by spatial proximity, not only the dependencies induced by proximity along the backbone.

There are many papers using spatial neighborhood information of residues to predict-protein interaction sites (see e.g. [2,13,21,28]). However, the spatial information of proteins was only integrated into the feature functions, but *not* represented in the model. For probabilistic models, the difference between the two ways to integrate spacial information is that in previous models the label of the i-th residue Y_i is conditional independent from the labels of other residues given data $\mathbf{X}$ and – in the case of linear CRFs or HMMs – given the labels of Y_{i-1} and Y_{i+1}. Even when neighborhood information is only used for spatial smoothing of the labels, the intuitive advantage over, say, an SVM classifier that uses spatial neighborhood *in the features* but classifies each residue *independently*, is that not-patch-like candidate labelings are explicitly punished. In contrast, such an independent classifier-approach may have a tendency to predict individual interface residues 'sprinkled' around the protein surface [28].

For this reason, a general CRF seems to be more suitable for the task. However, inference for general CRFs

is intractable. In this paper, pairwise conditional random fields (pCRFs) are utilized. Specializing general CRFs, only node cliques and edge cliques are taken into consideration in pCRFs. A pCRF retains most spatial information of proteins, can be specified with the same number of parameter as a line CRF and approximate inference remains feasible with the generalization of the Viterbi algorithm introduced here. Taking pattern from piecewise training methods [35], we disentangled the labels of nodes and edges to train the model.

In order to take advantage of a residue-wise score-based predictor, we model the protein surface by means of a pCRF, where the observation is solely a sequence of surface residue scores between 0 and 1 output by the predictor. We then utilize a generalized Viterbi algorithm and piecewise training. The resulting tool tries to enhance the predictor chosen on the surface of the protein under study. It is the aim of this paper to demonstrate effectiveness of this approach provided that the interface residue scores and the non-interface residue scores are appropriately distributed.

Methods

We address the problem of improving residue-wise score-based predictors for protein interface residues as a node labeling problem for undirected graphs using the model class of conditional random fields (CRFs). Lafferty *et al.* [30] were the first who applied CRFs to the problem of labeling sequence data. Li *et al.* [28] used line CRFs to address the interaction site prediction. They have the advantage that the Viterbi algorithm well-known from decoding HMMs can be used to efficiently infer the most likely labeling sequence. Very useful and illustrative presentations on CRFs are given in [31,32,36,37]. Above CRF-based models make the assumption that the label of one residue is conditionally independent of the labels of all other residues given the labels of the two adjacent residues in the protein sequence. To the best of our knowledge, we are the first to employ a graphical model that takes the spatial neighborhood of residues located on the protein surface into account.

This section is subdivided into three parts. We first explain how we model protein surfaces by pairwise CRFs. Then we introduce our new inference method. Finally, we elucidate our training method.

Using conditional random fields to model protein surfaces

For every protein under study that has n surface residues, a pair of random vectors $(\mathbf{X}, \mathbf{Y})$ is considered. The vector $\mathbf{X}$ is the *observation* that represents the knowledge about this protein that is utilized in the prediction, e.g. the 3D structure of the target protein and a multiple sequence alignment together with homologs.

The vector $\mathbf{Y}$ is a random sequence of length n over the alphabet $\{\mathrm{I}, \mathrm{N}\}$ that labels the index set $\{1, 2, \ldots, n\}$, which in turn is called the set of *positions* (of the surface residues). The label I represents interface residues, whereas the label N represents non-interface residues. $\{\mathrm{I}, \mathrm{N}\}^n$ is the set of all label sequences of length n over $\{\mathrm{I}, \mathrm{N}\}$. We will also call them *assignments* as the term 'label sequence' may lead to confusion when applied below to subsets of $\{1, 2, \ldots, n\}$ that are not contiguous sequences.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the *neighborhood graph*, where $\mathcal{V} = \{1, 2, \ldots, n\}$ is the set of positions, $\mathcal{E}$ is the set of edges that typically results from an atom-distance-based neighborhood definition for positions. We assume for convenience in notation that $\mathcal{G}$ has no isolated nodes. Cases with isolated nodes could trivially be reduced to cases without isolated nodes. Let $\mathcal{C}$ be the set of $\mathcal{G}$'s *cliques*, which we refer to as *node cliques*. For a node clique $c \in \mathcal{C}$ and an assignment $\mathbf{y}$ we denote by $\mathbf{y}_c$ the restriction of $\mathbf{y}$ to the positions belonging to the node clique c. For $c = \{i\}$ and $c = \{i, j\}$ we write y_i and (y_i, y_j) rather than $\mathbf{y}_{\{i\}}$ and $\mathbf{y}_{\{i,j\}}$.

The preceding notation is also used in the slightly more general case of partial label assignments to arbitrarily chosen subsets $\mathcal{S}$ of the set of positions $\mathcal{V}$. Formally, let $\mathbf{y}_\mathcal{S}$ denote $\{(i, y_i) \mid i \in \mathcal{S},\ y_i \in \{\mathrm{I}, \mathrm{N}\}\}$. Given two partial assignments $\mathbf{y}_{\mathcal{S}_1}$ and $\mathbf{y}_{\mathcal{S}_2}$ are identical on $\mathcal{S}_1 \cap \mathcal{S}_2$, the union $\mathbf{y}_{\mathcal{S}_1} \cup \mathbf{y}_{\mathcal{S}_2}$ is well-defined.

The conditional distribution function of our pCRF $(\mathbf{X}, \mathbf{Y})$ with respect to the neighborhood graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is defined as follows:

$$\mathbb{P}(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left(\sum_{i\in\mathcal{V}} \Phi_i(y_i, \mathbf{x}) + \sum_{\{i,j\}\in\mathcal{E}} \Phi_{i,j}(y_i, y_j, \mathbf{x}) + \sum_{c\in\mathcal{C}\setminus(\mathcal{V}\cup\mathcal{E})} \Phi_c(\mathbf{y}_c, \mathbf{x})\right), \quad (1)$$

where $\mathbf{x}$ and $\mathbf{y}$ are arbitrarily chosen instances of the random observation $\mathbf{X}$ and the random label sequence $\mathbf{Y}$, respectively, $\Phi_c(y_c, \mathbf{x}) \in \bullet\ (c \in \mathcal{C})$ is the feature of the CRF located at the node clique c (again Φ_i and $\Phi_{i,j}$ simplify notation for $\Phi_{\{i\}}$ and $\Phi_{\{i,j\}}$), and $Z(\mathbf{x})$ is the observation-specific *normalization factor* defined by

$$Z(\mathbf{x}) := \sum_{\mathbf{y}\in\{\mathrm{I},\mathrm{N}\}^n} \exp\left(\sum_{i\in\mathcal{V}} \Phi_i(y_i, \mathbf{x}) + \sum_{\{i,j\}\in\mathcal{E}} \Phi_{i,j}(y_i, y_j, \mathbf{x}) + \sum_{c\in\mathcal{C}\setminus(\mathcal{V}\cup\mathcal{E})} \Phi_c(\mathbf{y}_c, \mathbf{x})\right). \quad (2)$$

Let us call $\ln(Z(\mathbf{x})\mathbb{P}(\mathbf{y}\,|\,\mathbf{x}))$ the *score* of the label sequence $\mathbf{y}$ given the observation $\mathbf{x}$.

A CRF is called a *pairwise CRF* (pCRF) if $\Phi_c \equiv 0$, for all node cliques c larger than two. The remaining features Φ_i and $\Phi_{i,j}$ are referred to as *node features*

and *edge features*, respectively. Thus, every position $i \in \mathcal{V}$ and every edge $(i,j) \in \mathcal{E}$ is represented by the pair $(\Phi_i(\mathrm{N},\mathbf{x}), \Phi_i(\mathrm{I},\mathbf{x}))$ and by the quadruplet $\big(\Phi_{\{i,j\}}(\mathrm{N},\mathrm{N},\mathbf{x}), \Phi_{\{i,j\}}(\mathrm{I},\mathrm{N},\mathbf{x}), \Phi_{\{i,j\}}(\mathrm{N},\mathrm{I},\mathbf{x}), \Phi_{\{i,j\}}(\mathrm{I},\mathrm{I},\mathbf{x})\big)$.

Following [30], we assume moreover that each node feature and each edge feature is a sum of weighted base features. More precisely, for every position $i \in \mathcal{V}$ and every edge $\{i,j\} \in \mathcal{E}$ we assume representations

$$\Phi_i(y_i,\mathbf{x}) = \sum_{k=1}^{K_1} \alpha_k \psi_k(i,y_i,\mathbf{x})$$

$$\Phi_{i,j}(y_i,y_j,\mathbf{x}) = \sum_{k=1}^{K_2} \beta_k \phi_k(i,j,y_i,y_j,\mathbf{x}),$$

where $\mathbf{y} \in \{\mathrm{I},\mathrm{N}\}^n$ and $\mathbf{x}$ is an observation. The two real vectors

$$\alpha := \big(\alpha_1,\alpha_2,\ldots,\alpha_{K_1}\big) \quad \beta := \big(\beta_1,\beta_2,\ldots,\beta_{K_2}\big) \tag{3}$$

need to be calculated in a training phase.

In the most general sense, protein characteristics are real-valued evaluations of positions and pairs of adjacent positions (edges of the neighborhood graph), respectively, that are correlated with our position labeling problem. We use a standard step function technique to obtain base features from protein characteristics, rather than taking the raw values of the characteristics. To make our paper self-contained, let us describe this technique for short.

A protein characteristic depends on the observation and either a node or an edge. Each protein characteristic, such as e.g. the relative solvent-accessible surface area of a residue, is transformed into several binary features by binning, i.e. we distinguish only a few different cases rather than the whole range of the characteristic. Assuming the common case of real-valued characteristics, the bins are a partition of the reals into intervals. The use of this discretization allows to approximate any shape of dependency of the labels on the characteristics, rather than assuming a fixed shape such as linear or logarithmic.

From protein characteristics for positions to node features. We subdivide the range of the characteristics C into say γ intervals, where γ is at least two. Let $s_1 < s_2 < \ldots < s_{\gamma-1}$ be the corresponding interval boundaries. It is reasonable to take s_ι as the ι/γ-quantile of the empirical distribution of C for non-interface residues, where $C(i,\mathbf{x}) \in (s_0, s_\gamma]$. Then we define for each position $i \in \mathcal{V}$ the following 2γ base features associated with the position characteristics C.

$$\phi^{(C)}_{y,\iota}(i,y_i,\mathbf{x}) := \begin{cases} 1 & \text{if } y_i = y \text{ and } C(i,\mathbf{x}) \in (s_\iota, s_{\iota+1}]\,; \\ 0 & \text{otherwise;} \end{cases} \tag{4}$$

where $y = \mathrm{N},\mathrm{I}$, and $\iota = 0,1,\ldots,\gamma-1$ and $s_0 := -\infty, s_\gamma := \infty$.

From protein characteristics for edges to edge features. Let D be the characteristics. Analogous to the previous case, we then obtain for each edge $\{i,j\} \in \mathcal{E}$ the following 4γ base features associated with D, where $y, y' \in \{\mathrm{N},\mathrm{I}\}$ and $\iota = 0,1,\ldots,\gamma-1$.

$$\phi^{(D)}_{y,y',\iota}(i,j,y_i,y_j,\mathbf{x}) := \begin{cases} 1 \text{ if } y_i = y, y_j = y', \text{and } D(i,j,\mathbf{x}) \in (s_\iota, s_{\iota+1}]; \\ 0 \text{ otherwise.} \end{cases} \tag{5}$$

In both cases we set $\gamma = 5$.

Devising a generalized Viterbi algorithm for pCRFs

The problem of finding a most probable label sequence $\mathbf{y}^*$ given an observation $\mathbf{x}$ is **NP**-hard for general pCRFs [31]. In this subsection we present a heuristic that approximately solves this problem.

To this end, we first devise an algorithm, which we call *generalized Viterbi algorithm*. It computes an optimal label sequence, where the posterior probability of $\mathbf{y}^*$ given $\mathbf{x}$ is maximized. Unfortunately, its run-time is in too many cases not acceptable. That is why we transform it in a second step into a feasible, time-bounded approximation algorithm.

The generalized Viterbi algorithm

Let $\mathcal{G} = (\mathcal{V},\mathcal{E})$ be the neighborhood graph underlying the protein under study. For any assignment (label sequence) $\mathbf{y}$ and any subset $\mathcal{V}'$ of $\mathcal{V}$, let $\mathbf{y}_{\mathcal{V}'}$ denote the partial assignment of $\mathbf{y}$ with respect to $\mathcal{V}'$. (This is in line with the notation $\mathbf{y}_c$ (c a position clique) introduced earlier in this study).

If $\mathcal{V}_1, \mathcal{V}_2, \ldots, \mathcal{V}_r$ are pairwise disjoint position sets, the assignment for $\mathcal{V}_1 \cup \mathcal{V}_2 \cup \ldots \cup \mathcal{V}_r$ canonically resulting from assignments $\mathbf{y}_{\mathcal{V}_1}, \mathbf{y}_{\mathcal{V}_2}, \ldots, \mathbf{y}_{\mathcal{V}_r}$ is denoted by $\mathbf{y}_{\mathcal{V}_1} \cup \mathbf{y}_{\mathcal{V}_2} \cup \ldots \cup \mathbf{y}_{\mathcal{V}_r}$. For $\mathcal{V}' \subset \mathcal{V}$, the score $\mathrm{s}_{\mathcal{V}'}(\mathbf{y}_{\mathcal{V}'} \,|\, \mathbf{x})$ is defined by

$$\mathrm{s}_{\mathcal{V}'}(\mathbf{y}_{\mathcal{V}'} \,|\, \mathbf{x}) := \sum_{i \in \mathcal{V}'} \Phi_i\,(y_i,\mathbf{x}) + \sum_{\substack{i,j \in \mathcal{V}' \\ \{i,j\} \in \mathcal{E}}} \Phi_{i,j}\,\big(y_i,y_j,\mathbf{x}\big)\,.$$

Then the problem of determining a most probable label sequence $\mathbf{y}^*$ given an observation $\mathbf{x}$ can be reformulated as

$$\mathbf{y}^* = \operatorname*{argmax}_{\mathbf{y}}\ \mathrm{s}_{\mathcal{V}}(\mathbf{y} \,|\, \mathbf{x}).$$

This is the case, because it suffices to consider the score.

To put this into practice, we devised an algorithm we call generalized Viterbi. On the one hand, it is analogous to the classical Viterbi algorithm. On the other hand, there is a major difference. In our case there is no canonical order in which the positions of $\mathcal{G}$ are traversed. Having explained our algorithm for any order, we show how to

calculate a fairly effective one. In what follows, we assume that the positions not yet touched are held in a dynamic queue. Those positions having already left the queue form the *history set* $\mathcal{H} \subseteq \mathcal{V}$.

Assume that the subgraph of $\mathcal{G}$ induced by $\mathcal{H}$ has connected components $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_m$. For $\mu = 1, 2, \ldots, m$, let $\mathcal{B}_\mu \subseteq \mathcal{H}_\mu$ be the so-called *boundary component* associated with $\mathcal{H}_\mu$ defined by $\mathcal{B}_\mu := \left\{ i \in \mathcal{H}_\mu \mid \exists\, j \notin \mathcal{H}, \{i,j\} \in \mathcal{E} \right\}$. The complement $\mathcal{H}_\mu \setminus \mathcal{B}_\mu$ is the *interior* of the μ-th history component. See Figure 1 for an example.

For assignments $\mathbf{y}_{\mathcal{B}_1}, \mathbf{y}_{\mathcal{B}_2}, \ldots, \mathbf{y}_{\mathcal{B}_m}$ of the boundary components $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_m$, the Viterbi variables $\mathrm{vit}_{\mathcal{H}_1}(\mathbf{y}_{\mathcal{B}_1}), \mathrm{vit}_{\mathcal{H}_2}(\mathbf{y}_{\mathcal{B}_2}), \ldots, \mathrm{vit}_{\mathcal{H}_m}(\mathbf{y}_{\mathcal{B}_m})$ are defined as

$$\mathrm{vit}_{\mathcal{H}_\mu}\left(\mathbf{y}_{\mathcal{B}_\mu}\right) := \max_{\mathbf{y}_{\mathcal{H}_\mu \setminus \mathcal{B}_\mu}} \mathrm{s}_{\mathcal{H}_\mu}\left(\mathbf{y}_{\mathcal{H}_\mu \setminus \mathcal{B}_\mu} \cup \mathbf{y}_{\mathcal{B}_\mu} \mid \mathbf{x}\right) \times (\mu = 1, 2, \ldots, m). \tag{6}$$

The Viterbi variables can be represented as a set of tables, one table of size $2^{|\mathcal{B}_\mu|}$ for each boundary component $\mathcal{B}_\mu$. In the case where a boundary component is empty the table reduces to a single number.

At any stage, the algorithm stores the connected components $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_m$ of the current history set $\mathcal{H}$, the corresponding boundary components $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_m$, and Viterbi variable values $\mathrm{vit}_{\mathcal{H}_1}(\mathbf{y}_{\mathcal{B}_1}), \mathrm{vit}_{\mathcal{H}_2}(\mathbf{y}_{\mathcal{B}_2}), \ldots, \mathrm{vit}_{\mathcal{H}_m}(\mathbf{y}_{\mathcal{B}_m})$, where $\mathbf{y}_{\mathcal{B}_1}, \mathbf{y}_{\mathcal{B}_2}, \ldots, \mathbf{y}_{\mathcal{B}_m}$ range over all possible assignments of corresponding boundary component. We store for every assignment on the boundary, a maximizing interior assignment. This assignment is the argmax of (6) but is determined with the dynamic programming recursions defined below. Let us call these data the current state of the algorithm. It mainly consists of record sets indexed by the boundary labelings.

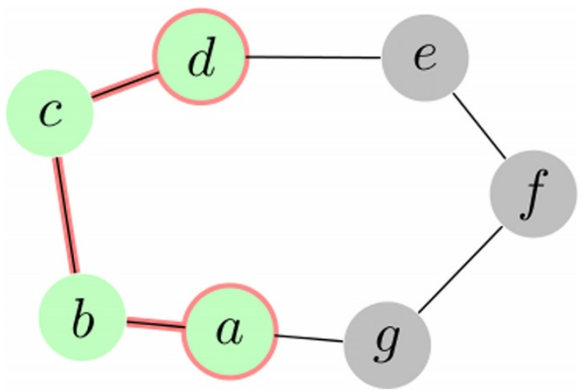

Figure 1 Example history set $\mathcal{H} = \{a, b, c, d\}$ having boundary $\mathcal{B} = \{a, d\}$.

At the very beginning the queue contains all positions, the history set $\mathcal{H}$ and the corresponding boundary component $\mathcal{B}$ are empty. As long as the position queue is not empty, the top element v is extracted and the state is updated as follows.

Adjoining v to the history set $\mathcal{H}$, there are two cases to distinguish. Either position v is not adjacent to any other position of any old boundary component (see Figure 2) or adjoining position v to $\mathcal{H}$ results in adding it to some connected component of the old history set or even merging together two or more of them (see Figure 3).

In the first case we simply have to take over all the old connected components, boundary sets and Viterbi variables. Moreover, we perform the instructions

$$\mathcal{H}_{m+1} \leftarrow \mathcal{B}_{m+1} \leftarrow \{v\}, \quad \mathrm{vit}_{\mathcal{H}_{m+1}}(\mathrm{N}) \leftarrow \mathrm{s}_{\mathcal{H}_{m+1}}(\mathrm{N} \mid \mathbf{x}), \quad \mathrm{vit}_{\mathcal{H}_{m+1}}(\mathrm{I}) \leftarrow \mathrm{s}_{\mathcal{H}_{m+1}}(\mathrm{I} \mid \mathbf{x}).$$

In the second case position v is adjacent to some boundary components, say $\mathcal{B}_{m'}, \mathcal{B}_{m'+1}, \ldots, \mathcal{B}_m$. Then the old history components $\mathcal{H}_{m'}, \mathcal{H}_{m'+1}, \ldots, \mathcal{H}_m$ and the current position v are merged together:

$$\mathcal{H}_{tmp} \leftarrow \mathcal{H}_{m'} \cup \mathcal{H}_{m'+1} \cup \ldots \cup \mathcal{H}_m \cup \{v\}.$$

The other history set components and corresponding Viterbi variables are not affected.

For $\mu = m', m'+1, \ldots, m$, let $\mathcal{R}_\mu \subseteq \mathcal{B}_\mu$ be the set of all positions out of $\mathcal{B}_\mu$ that are no longer boundary nodes after having adjoined v to the history set. The nodes in $\mathcal{R}_\mu$ are removed from the boundary $\mathcal{B}_\mu$ after the iteration. Let $\tilde{\mathcal{B}}_\mu$ be the complement of $\mathcal{R}_\mu$ in $\mathcal{B}_\mu$. By inspecting the edges incident to the current position v, all these sets can be computed in linear time.

The new boundary set $\mathcal{B}_{tmp}$ is then either $\tilde{\mathcal{B}}_{m'} \cup \tilde{\mathcal{B}}_{m'+1} \cup \ldots \cup \tilde{\mathcal{B}}_m$ or $\tilde{\mathcal{B}}_{m'} \cup \tilde{\mathcal{B}}_{m'+1} \cup \ldots \cup \tilde{\mathcal{B}}_m \cup \{v\}$, where it can be checked in linear time whether or not v is a new boundary position.

We are now in a position to calculate the new Viterbi variables $\mathrm{vit}_{\mathcal{H}_{tmp}}(\mathbf{y}_{\mathcal{B}_{tmp}})$, where $\mathbf{y}_{\mathcal{B}_{tmp}}$ ranges over all assignments of the new boundary set $\mathcal{B}_{tmp}$.

If $v \notin \mathcal{B}_{tmp}$ then

$$\mathrm{vit}_{\mathcal{H}_{tmp}}\left(\mathbf{y}_{\mathcal{B}_{tmp}}\right) \leftarrow \max_{y_v} \left\{ \Phi_v(y_v, \mathbf{x}) + \sum_{\mu=m'}^{m} \max_{\mathbf{y}_{\mathcal{R}_\mu}} \times \left(\mathrm{vit}_{\mathcal{H}_\mu} \mathbf{y}_{\tilde{\mathcal{B}}_\mu} \cup \mathbf{y}_{\mathcal{R}_\mu} + \sum_{\substack{w \in \mathcal{B}_\mu \\ \{v,w\} \in \mathcal{E}}} \Phi_{v,w}(y_v, y_w, \mathbf{x}) \right) \right\}.$$

Here, any assignment of a node set is assumed to implicitly define assignments for any subset thereof. Figure 4

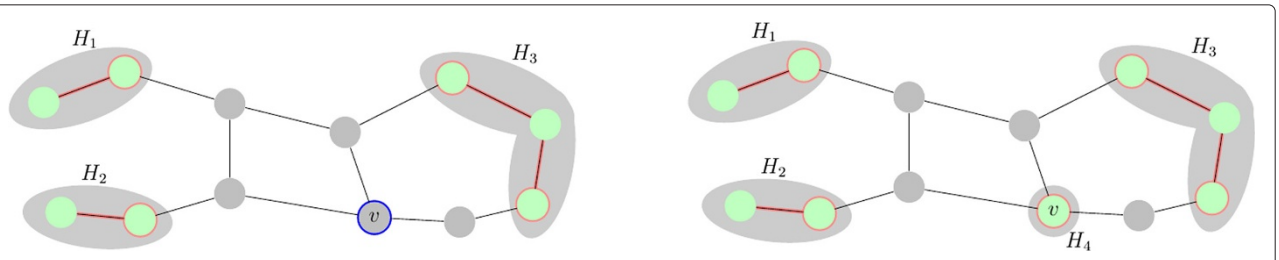

Figure 2 Computing the connected components of the new history set - case one.

illustrates this case of the recursion step. If, however, $v \in \mathcal{B}_{tmp}$, then

$$\begin{aligned}\mathrm{vit}_{\mathcal{H}_{tmp}}\left(\mathbf{y}_{\mathcal{B}_{tmp}}\right) \leftarrow \Phi_v\left(y_v, \mathbf{x}\right) &+ \sum_{\substack{w \in \tilde{\mathcal{B}}_{\mu'} \cup \cdots \cup \tilde{\mathcal{B}}_{\mu} \\ (v,w) \in \mathcal{E}}} \Phi_{v,w}\left(y_v, y_w, \mathbf{x}\right) \\ &+ \sum_{\mu=m'}^{m} \left\{ \max_{\mathbf{y}_{\mathcal{R}_\mu}} \left(\mathrm{vit}_{\mathcal{H}_\mu}\left(\mathbf{y}_{\tilde{\mathcal{B}}_\mu} \cup \mathbf{y}_{\mathcal{R}_\mu}\right) \right.\right. \\ &+ \left.\left. \sum_{\substack{w \in \mathcal{R}_\mu \\ (v,w) \in \mathcal{E}}} \Phi_{v,w}\left(y_v, y_w, \mathbf{x}\right) \right) \right\}.\end{aligned}$$

Finally, the interior labeling is stored, where the maximum is attained. The algorithm terminates after the last node v from $\mathcal{V}$ has been processed. In the typical case, where the graph is connected, at termination $m = 1, \mathcal{H}_1 = \mathcal{V}, \mathcal{B}_1 = \emptyset$.

The running time of the algorithm is $\mathcal{O}(n2^b)$, where b is the size of the largest boundary set and n is the number of surface residues. We call this algorithm *generalized Viterbi* algorithm as for the case of a graph that is a linear chain $1 - 2 - 3 - \cdots - n$ of nodes using the node order $1, 2, \ldots, n$ the Viterbi variables we define are the same as in the standard Viterbi algorithm for HMMs. In the case of a graph that is a tree, this algorithm specializes to the Fitch algorithm or an argmax-version of Felsenstein's pruning algorithm when a leaf-to-root node order is chosen after rooting the tree at an arbitrary node. In both special cases the boundary sets always have size at most 1. The tree example also motivate the use of several history sets at the same time: using a single history set only, one would not be able to achieve a linear running time on trees.

A heuristic based on the generalized Viterbi algorithm

First, it is vital for our generalized Viterbi algorithm to keep the size of the boundary sets small. A good position order is here of great importance. The algorithm starts by choosing a vertex of minimal degree. When determining the next position to be dequeued, the algorithm selects a boundary node such that the number of incident edges leading to nodes not belonging to any current history set is minimal. In an arbitrarily chosen order these nodes are dequeued next.

Second, the space demand is reduced by restricting the number of boundary labelings admitted. Starting from the available labelings of the current history set, the percentage of the reachable boundary labelings of the successor history that will be discarded is calculated. Then the corresponding percentile is estimated. To this end, a sufficiently large sample of possible labelings of the new boundary set is drawn, the Viterbi variables are computed, and the corresponding sample percentile is taken. Finally, only those boundary labelings of the new history set are retained whose Viterbi variables exceed this percentile.

That way we compute near-optimal solutions good enough for our purposes within feasible computation time.

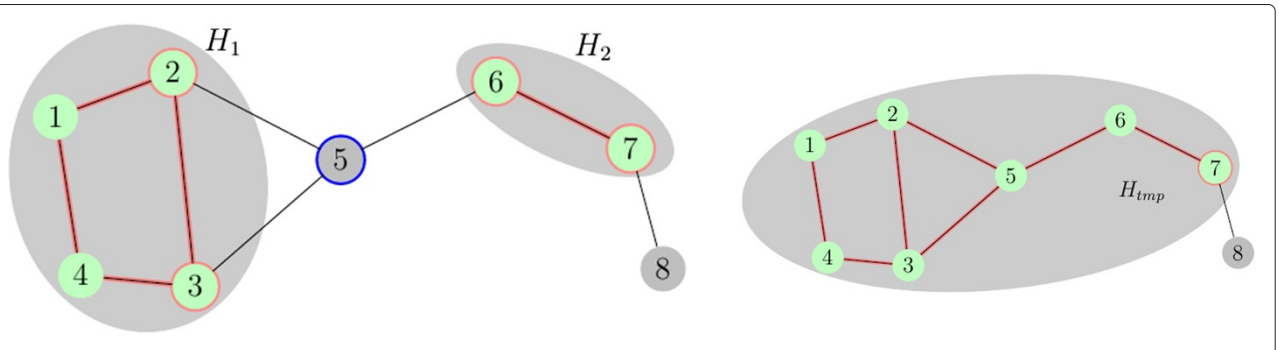

Figure 3 Computing the connected components of the new history set - case two.

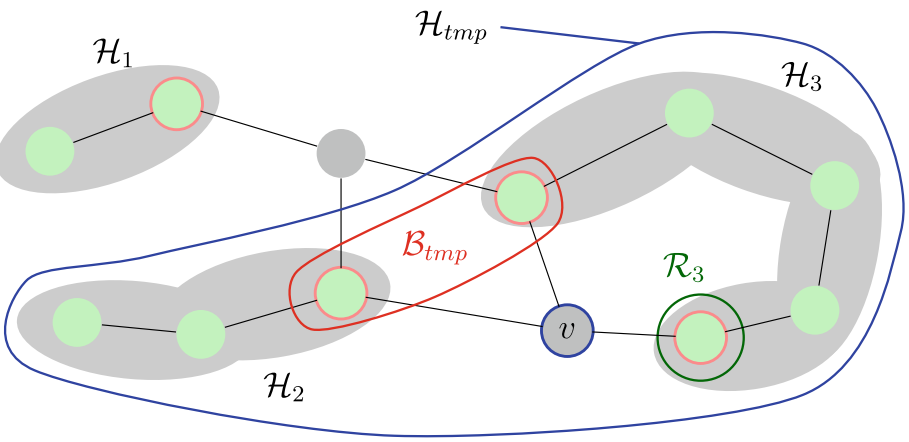

Figure 4 Viterbi recursion step in case $v \notin \mathcal{B}_{tmp}$. After adding node v to the history set, $\mathcal{H}_2$ and $\mathcal{H}_3$ will be replaced by $\mathcal{H}_{tmp} = \mathcal{H}_2 \cup \mathcal{H}_3 \cup \{v\}$. In this example, for every assignment of the new boundary $\mathcal{B}_{tmp}$ the score of $\mathcal{H}_{tmp}$ is maximized by varying over the assignments of v and $\mathcal{R}_3$ and using the Viterbi variables of $\mathcal{H}_2$ and $\mathcal{H}_3$.

Piecewise training for pCRFs

Let

$$\mathcal{D} := \left(\left(\mathbf{x}^{(1)}, \mathbf{y}^{(1)}\right), \left(\mathbf{x}^{(2)}, \mathbf{y}^{(2)}\right), \ldots, \left(\mathbf{x}^{(m)}, \mathbf{y}^{(m)}\right)\right)$$

be the independent identically distributed training sample. For every $\mu = 1, 2, \ldots, m$, let $\mathcal{V}_\mu$ and $\mathcal{E}_\mu$ be the set of positions and edges in the neighborhood graph associated with $\mathbf{x}_\mu$, let $n_\mu = |\mathcal{V}_\mu|$ be the number of positions of the μ-th training example and let $\{\mathrm{I}, \mathrm{N}\}^{n_\mu}$ be the set of all possible label sequences of this graph.

This data set is unbalanced as there are many more non-interface positions as interface positions. As customary for other machine learning approaches such as support vector machines and artificial neural networks [28], we here manipulated the ratio of positive and negative example positions for training in order to obtain reasonable results.

We have amplified the influence of the positive examples, rather than selecting various sets of training data by deleting negative ones as done in [28].

Let ν_I, ν_N, ν_{II} and ν_{NN} be the number of interface positions, the number of non-interface positions, the number of interface-interface edges, and the number of non-interface-non-interface edges in $\mathcal{D}$, respectively. Then we define the following two *amplifier functions* for all positions i and for all edges $\{i, j\}$ of the m neighborhood graphs resulting from the training data $\mathcal{D}$.

$$\eta_1(i) := \begin{cases} \frac{\nu_N}{\nu_I} - 1 & \text{if } y_i = \mathrm{I}; \\ 0 & \text{if } y_i = \mathrm{N}. \end{cases}$$

$$\eta_2(i,j) := \begin{cases} \frac{\nu_{NN}}{\nu_{II}} - 1 & \text{if } y_i = y_j = \mathrm{I}; \\ \frac{\nu_N}{\nu_I} - 1 & \text{if } y_i \neq y_j; \\ 0 & \text{if } y_i = y_j = \mathrm{N}. \end{cases}$$

To uniformly govern the influence of the amplifiers, we introduce an *amplifier control parameter* $\eta_3 \in [0, 1]$.

We set up our two log-likelihood objective function by

$$\begin{aligned} \ell\left(\lambda^{(\mathbf{s})}, \eta_3\right) := & \sum_{\mu=1}^{m} \sum_{i \in \mathcal{V}_\mu} (1 + \eta_3 \eta_1(i)) \sum_{k=1}^{K_1} \alpha_k \psi_k\left(i, y_i^{(\mu)}, \mathbf{x}^{(\mu)}\right) \\ & + \sum_{\mu=1}^{m} \sum_{\{i,j\} \in \mathcal{E}_\mu} \left(1 + \eta_3 \eta_2(i,j)\right) \sum_{k=1}^{K_2} \beta_k \phi_k \\ & \times \left(i, j, y_i^{(\mu)}, y_j^{(\mu)}, \mathbf{x}^{(\mu)}\right) - \sum_{\mu=1}^{m} \ln Z\left(\mathbf{x}^{(\mu)}, \eta_3\right), \end{aligned}$$

where ideally for each $\mu = 1, 2, \ldots, m$

$$\begin{aligned} Z\left(\mathbf{x}^{(\mu)}, \eta_3\right) := & \sum_{\mathbf{y} \in \{\mathrm{I},\mathrm{N}\}^{n_\mu}} \exp\left(\sum_{i \in \mathcal{V}_\mu} (1 + \eta_3 \eta_1(i)) \sum_{k=1}^{K_1} \alpha_k \psi_k\left(i, y_i, \mathbf{x}^{(\mu)}\right) \right. \\ & \left. + \sum_{\{i,j\} \in \mathcal{E}_\mu} \left(1 + \eta_3 \eta_2(i,j)\right) \sum_{k=1}^{K_2} \beta_k \phi_k\left(i, j, y_i, y_j, \mathbf{x}^{(\mu)}\right) \right) \end{aligned}$$

is the training-instance-specific normalization factor.

Unfortunately, maximizing this objective function in general is algorithmically intractable. Taking pattern from Sutton *et al.* [35] who introduced what they called piecewise training, we deal with this problem by disentangling the labels of nodes and edges. For $\mu = 1, 2, \ldots, m$, a *non-coherent labeling* $\mathbf{y} \in \{\mathrm{I}, \mathrm{N}\}^{\mathcal{V}_\mu \times \mathcal{E}_\mu}$ of the neighborhood graph $\left(\mathcal{V}_\mu, \mathcal{E}_\mu\right)$ is any mapping that assigns to every position $v \in \mathcal{V}_\mu$ and every edge $e \in \mathcal{E}_\mu$ a label $\mathbf{y}_v \in \{\mathrm{I}, \mathrm{N}\}$ and a pair of labels $\mathbf{y}_e \in \{\mathrm{I}, \mathrm{N}\}^2$, respectively.

We then replace $Z\left(\mathbf{x}^{(\mu)}, \eta_3\right)$ by

$$\tilde{Z}\left(\mathbf{x}^{(\mu)}, \eta_3\right) := \sum_{\mathbf{y}\in\{\mathrm{I,N}\}^{\mathcal{V}_\mu \times \mathcal{E}_\mu}} \exp\left(\sum_{v\in\mathcal{V}_\mu} (1+\eta_3\eta_1(v)) \sum_{k=1}^{K_1} \alpha_k \psi_k \times \left(v, \mathbf{y}_v, \mathbf{x}^{(\mu)}\right) + \sum_{e\in\mathcal{E}_\mu} (1+\eta_3\eta_2(e)) \sum_{k=1}^{K_2} \beta_k \phi_k \times \left(e, \mathbf{y}_e, \mathbf{x}^{(\mu)}\right)\right)$$

as normalization factor. This makes the optimization problem computationally feasible.

The L-BFGS method [38] is used to solve it. That way we obtain the coefficient vectors α and β (see Equations 3), which depend on the amplifier control parameter $\eta_3 \in [0, 1]$.

To mitigate the negative consequences of disentanglement, we use a *correction factor* $\delta \geq 1$. For any characteristics D and $\iota = 0, 1, \ldots, \gamma - 1$, the weights of the bases edge features $\phi^{(D)}_{\mathrm{I,I},\iota}$ and $\phi^{(D)}_{\mathrm{N,N},\iota}$ (see Equation 5) are all multiplied by δ. Thus a change in classification along an edge is additionally penalized. The correction factor δ is set best between 1.15 and 1.25.

For our implementation of the training, we used the Java CRF package from Sunita Sarawagi at http://crf.sourceforge.net/.

Results and discussion

In this section we demonstrate effectiveness of our pCRF-based protein surface model to enhance residue-wise score-based predictions of protein-protein interfaces. For the sake of ensuring reliability of the methods we used three data sets. The first one is *PlaneDimers* due to Zellner *et al.* [25], the second one is the list of 1276 two-chain-proteins published by Keskin *et al.* [39], which was used by Li*et al.* [28] to test their linear-chain CRF. Third, we used a non-redundant data set containing 22604 unique interface structures very recently compiled by Cukuroglu *et al.* and published in [40].

The data set *PlaneDimers* is less known than the data due to Keskin *et al.*. It consists of redundancy-free homodimers with flat protein-protein interfaces. Zellner *et al.* [25] developed an SVM, called *PresCont*, that assigns to each residue on the protein surface a score between 0 and 1, which we refer to as *PresCont* score in the sequel. The larger the score, the more likely the residue belongs to the interface. Zellner *et al.* made the prediction by thresholding the score. The *PresCont* server and the data list *PlaneDimers* are publicly available (see http://www-bioinf.uni-regensburg.de/).

In the first subsection we describe two sets of experiments performed with synthetic data, one on *PlaneDimers* [25], the other one on the list published by Keskin *et al.* [39]. In both cases we independently assign to each surface position a random score drawn according to two different parametrized sequences of β-distributions $\mathrm{Beta}(\alpha_\mathrm{I}(\varsigma)\beta_\mathrm{I}(\varsigma))$ and $\mathrm{Beta}(\alpha_\mathrm{N}(\varsigma)\beta_\mathrm{N}(\varsigma))$, one for the interface sites determined by the reference labeling, the other one for the non-interface positions. The parametrized values $\alpha_\mathrm{I}(\varsigma)$, $\alpha_\mathrm{N}(\varsigma)$, $\beta_\mathrm{I}(\varsigma)$ and $\beta_\mathrm{N}(\varsigma)$ determining the two sequences of distributions are chosen such that the following conditions are satisfied. The mean values $e_\mathrm{I} > e_\mathrm{N}$ are the average *PresCont* scores on interface sites and non-interface sites of all chains from *PlaneDimers*. The variances σ^2_I and σ^2_N are equal to $\sigma^2_{\mathrm{I},0}\varsigma$ and $\sigma^2_{\mathrm{N}.0}\varsigma$, where $\sigma^2_{\mathrm{I},0}$ and $\sigma^2_{\mathrm{N}.0}$ are the corresponding variances of the *PresCont* score, and $\varsigma \in \{0.8, 0.9, 1.0, 1.1, 1.2\}$ models the precision of the synthetic score. The deciding feature of all these distributions is that they are *unimodal*. The result of the subsection is that enhancement works for unimodal score distributions.

The second subsection is about a synthetic data experiment on a new data set due to Cukuroglu [40]. Here we follow the line of the first subsection except for the fact that we restrict ourselves to signal precision $\varsigma = 1.0$.

In the third subsection we study the *PresCont* scores for two-chain protein complexes from the data set *PlaneDimers*. According to Figure 5, the *PresCont* score for non-interface residues is far from being unimodal. However, if one restrict oneself to the part above a threshold in the neighborhood of 0.5 and larger, one may ask whether enhancement restricted to that domain will works. The subsection answers this question in the affirmative. Having chosen a threshold as described above, one can improve the classification with respect to this threshold as follows. Take over the prediction for scores below the threshold and reclassify the residues the scores of which are above by means of the pCRF-based enhancer.

In general, observations $\mathbf{x}$ could encompass a PDB file, which in particular determines the 3D-structure of the protein, together with an MSA that models evolutionary

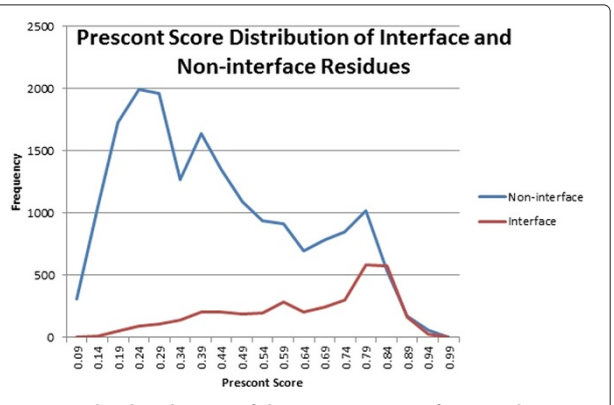

Figure 5 The distribution of the *PresCont* score for complexes from the data set *PlaneDimers*.

aspects. In our case an observation solely consists of the *PresCont* score sequence or of the sequence of synthetic scores for the surface residues. Formally, every observation $\mathbf{x}$ is equal to a vector $(\zeta_1, \zeta_2, \ldots, \zeta_n) \in [0,1]^n$.

There are several neighborhood notions for residues, surface/core definitions and interface determinations in the literature. When studying the data set *PlaneDimers*, we follow [25]. In the case of the list due to Keskin *et al.* [39], the definitions according to [28] are used. Finally, when studying complexes taken from the data set published in [40], we take the following definitions. The RASA value of a surface residue is at least 15% (see [28]). Two residues are defined as contacting if the distance between any two of their atoms is less than the sum of the corresponding van der Waals radii plus 0.5 Å (see [40]).

Anyway, according to Keskin *et al.* [39] we define the *distance* of two residues on one and the same chain as the distance of their major carbon atoms. We then say that one residue is *nearby* another residue, if they are at distance below 6 Å. (Note that usually residues adjacent on backbone are at distance of less than or equal to 3.5 Å). This definition in turn is the basis of the neighborhood graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ underlying the pCRF. Two surface positions are joined together by an undirected edge if and only if the corresponding residues are nearby ones.

Our pCRF-based enhancer utilizes one position characteristic and two edge characteristics on the basis of the standard step function method explained in the Methods section. If $\mathbf{x} = (\zeta_1, \zeta_2, \ldots, \zeta_n) \in [0,1]^n$ is the observation associated with the protein under study, and if $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is the neighborhood graph, then for every position $i \in \mathcal{V}$ and every edge $(i,j) \in \mathcal{E}$ we set

$$C(i,\mathbf{x}) := \zeta_i \quad D_1(i,j,\mathbf{x}) := \max\{\zeta_i, \zeta_j\} \quad D_2(i,j,\mathbf{x}) := |\zeta_i - \zeta_j|.$$

To enhance predictions obtained by thresholding, solely information coming from the residue neighborhood relations on the surface is additionally used.

In order to be able to calculate the performance measure of *area under the ROC curve* (AUC) for our pCRF-based enhancer on synthetic scores, we proceed as follows. For each edge $\{i,j\} \in \mathcal{E}$, we replace the local feature value $\Phi_{i,j}(\mathrm{I},\mathrm{I},\mathbf{x})$ by $\kappa\Phi_{i,j}(\mathrm{I},\mathrm{I},\mathbf{x})$, where $\kappa \in (0,\infty)$.

We enhance residue-wise score-based predictors only on the protein surface. In our synthetic data experiments there is no predictor available for core residues. For proteins taken from the data list published by Keskin *et al.* [39] it happens that interface sites belong to the core. That is why we use what we call *Surface AUC Ratio* Γ of the enhancer as our performance measure for our synthetic data experiments.

$$\Gamma := \frac{\text{AUC referred to the protein surface of the enhancer}}{\text{AUC referred to the protein surface of the residue-wise score-based threshold predictor}}.$$

If Γ is greater than 1, the enhancement was successful. The larger Γ, the greater success.

To estimate performance measures, we applied 5-fold cross-validation experiments.

A fully built-out pCRF-based tool box for modeling protein surfaces needs to comprise all the standard algorithms as e.g. forward-backward techniques, marginalization and posterior decoding known for HMMs and linear-chain CRFs. To begin with, in the fourth subsection we explain how to put a variant form of the forward algorithm and posterior decoding for pCRFs into practice.

Simulating unimodal scores of various precisions

We estimated means e_{I} and e_{N} and variances $\sigma^2_{0,\mathrm{I}}$ and $\sigma^2_{0,\mathrm{N}}$ of the *PresCont* score on interface sites and non-interface positions of *PlaneDimers*, respectively, as follows.

$$\begin{aligned} \hat{e}_{\mathrm{I}} &= 0.61488 \quad \hat{e}_{\mathrm{N}} = 0.40590 \\ \hat{\sigma}^2_{0,\mathrm{I}} &= 0.03991 \quad \hat{\sigma}^2_{0,\mathrm{N}} = 0.04006 \end{aligned} \tag{7}$$

We randomly chose 120 instances under the uniform distribution from the data set published by Keskin *et al.* [39] to perform our experiments. Let us refer to this set as *KL-subset* in the sequel. (It is accessible at http://ppicrf.informatik.uni-goettingen.de/index.html).

Zellner *et al.* [25] used the following determinations. A residue is defined to be part of the protein surface, if its relative solvent-accessible surface area is at least 5% [17]. A surface residue is said to constitute an *inter-facial contact*, if there exists at least one atom of this residue which has a van-der-Waals-sphere at a distance of at most 0.5 Å from the van-der-Waals sphere to any atom from a partner chain residue [39].

Based on [3,12,15,20,41], Li *et al.* [28] assume an inter-facial contact of a residue on a chain is assumed to be there, if any heavy atom of this residue is at distance of at most 5 Å from any heavy atom from a partner chain. The relative solvent-accessible surface area of surface residue is at least 15%.

We independently assigned to each interface surface residue of the two data sets a random score between zero and one according to the β-distribution $\mathrm{Beta}(\alpha_{\mathrm{I}}(\varsigma)\beta_{\mathrm{I}}(\varsigma))$, and to every non-interface surface residue a score according to $\mathrm{Beta}(\alpha_{\mathrm{N}}(\varsigma)\beta_{\mathrm{N}}(\varsigma))$, where the score precision ς satisfies

$$\varsigma \in \{0.8, 0.9, 1.0, 1.1, 1.2\}, \tag{8}$$

and the parameters $\alpha_{\mathrm{I}}(\varsigma), \beta_{\mathrm{I}}(\varsigma), \alpha_{\mathrm{N}}(\varsigma), \beta_{\mathrm{N}}(\varsigma)$ were chosen such that

$$\begin{aligned} \hat{e}_{\mathrm{I}} &= \frac{\alpha_{\mathrm{I}}(\varsigma)}{\alpha_{\mathrm{I}}(\varsigma)+\beta_{\mathrm{I}}(\varsigma)} \\ \hat{\sigma}^2_{0,\mathrm{I}}\varsigma &= \frac{\alpha_{\mathrm{I}}(\varsigma)\beta_{\mathrm{I}}(\varsigma)}{(\alpha_{\mathrm{I}}(\varsigma)+\beta_{\mathrm{I}}(\varsigma))^2\,(\alpha_{\mathrm{I}}(\varsigma)+\beta_{\mathrm{I}}(\varsigma)+1)} \end{aligned} \tag{9}$$

$$\begin{aligned} \hat{e}_{\mathrm{N}} &= \frac{\alpha_{\mathrm{N}}(\varsigma)}{\alpha_{\mathrm{N}}(\varsigma)+\beta_{\mathrm{N}}(\varsigma)} \\ \hat{\sigma}^2_{0,\mathrm{N}}\varsigma &= \frac{\alpha_{\mathrm{N}}(\varsigma)\beta_{\mathrm{N}}(\varsigma)}{(\alpha_{\mathrm{N}}(\varsigma)+\beta_{\mathrm{N}}(\varsigma))^2\,(\alpha_{\mathrm{N}}(\varsigma)+\beta_{\mathrm{I}}(\varsigma)+1)} \end{aligned} \tag{10}$$

The Surface AUC Ratios of the enhancer compared with the threshold predictor on *PlaneDimers* and the *KL*-subset are displayed in Table 1. There is an improvement of $8.4\% - 9.3\%$ on *PlaneDimers* and of $3.2\% - 5.0\%$ on the KL-subset.

Moreover, we compared individual classification results obtained by thresholding the scores with pCRF-based enhanced predictions. Because of the fact that the specificity of the threshold predictor can be easily changed by manipulating the threshold, we proceeded as follows. For every score precision, the pCRF-based enhancer has a well-defined specificity referred to the surface residues. We then chose the threshold such that the specificity of the threshold predictor is close to that of the enhancer. The results are shown in Table 2. The sensitivity is increased by $53\% - 67\%$ on the data set *PlaneDimers* and by $14\% - 22\%$ on the KL-subset.

Table 1 and Table 2 justify the following conclusion. Enhancing the threshold prediction by our pCRF works provided that the distributions of the interface scores as well as the non-interface scores are unimodal. The enhancement for the data set *PlaneDimers* is larger than for the KL-subset. This might be caused by the plain interface geometry of the complexes taken from *PlaneDimers*.

Utilizing the new data set due to Cukuroglu [40]

As in the case of the KL-subset, we randomly chose 60 dimers. We refer to the resulting list as *CGNK-subset*. Having assigned synthetic scores according to Equations 7, 9 and 10, where $\varsigma = 1.0$, we compared individual classification results obtained by thresholding the scores with pCRF-based enhanced predictions in exactly the same way as we did for the KL-subset. The results are shown in Table 3. The sensitivity is increased by 22%.

A main finding of Cukuroglu [40] relevant to protein-protein interface prediction is, that the average interface RASA value is greater than 40%. Since our method is designed to improve performance of a given residue-wise predictor, using this result is not in the scope of this paper. However, a CRF-based predictor integrating features for cliques of size greater than 2 is not beyond the range of current algorithmic capabilities. In such a model a feature set that discretizes the mean RASA value of cliques is promising.

Enhancing the *PresCont* server prediction on *PlaneDimers*

For the sake of completeness, we shortly review the residue characteristics used by *PresCont*.

Relative solvent-accessible surface area

For any residue a, the *solvent-accessible surface area* $\mathbf{asa}(a)$ can be computed by e.g. the software library BALL [42]. Most of the classifiers known from the literature utilize this characteristic (see [43]). For *PresCont* the *relative solvent-accessible surface area* according to

$$\mathbf{rasa}(a) := \frac{\mathbf{asa}(a)}{\mathbf{asa}_{\max}(a)} \tag{11}$$

is taken into operation, where $\mathbf{asa}_{\max}(a)$ is the maximally possible accessible surface area of residue a [44].

Hydrophobicity

Many interfaces possess a hydrophobic core surrounded by a ring of polar residues [45,46]. In order to reduce noise, in [25] the contribution of hydrophobic patches rather then the influence of individual residues is utilized.

Residue conservation

Measures of this type utilized in [25] are the Shannon entropy and the relative Shannon entropy of empirical residue distributions in MSA columns. As an alternative, empirical expectations of BLOSUM-based similarities are taken for them.

Table 1 Classification results on *PlaneDimers* and the KL-subset, where the β-distributions according to which the synthetic scores were drawn are defined by Equations 7, 8, 9 and 10

PlaneDimers	**Score precision ς**	**0.8**	**0.9**	**1.0**	**1.1**	**1.2**
	Surface AUC ratio Γ	1.084	1.091	1.093	1.089	1.093
Kl-subset	Signal precision ς	0.8	0.9	1.0	1.1	1.2
	Surface AUC ratio Γ	1.032	1.039	1.045	1.045	1.050

Depending on the variances determined by ς, the enhancer increases the AUC referred to the protein surface by 8.4%-9.3%. on *PlaneDimers*, and by 3.2%-5.0% on the KL-subset.

Table 2 Comparing the enhancer with the threshold classifier of approximately equal specificity on synthetic scores assigned to surface residues of protein complexes taken from the data set *PlaneDimers* and the KL-subset

Data Set	Score Precision ς	Classifier	Specificity	Sensitivity	MCC
PlaneDimer	0.8	Threshold Predictor	0.9672	0.2562	0.3253
		Enhancer	0.9666	0.4281	0.4911
	0.9	Threshold Predictor	0.9618	0.2556	0.3077
		Enhancer	0.9624	0.4086	0.4610
	1.0	Threshold Predictor	0.9611	0.2428	0.2912
		Enhancer	0.9612	0.3872	0.4379
	1.1	Threshold Predictor	0.9681	0.2100	0.2753
		Enhancer	0.9677	0.3307	0.4045
	1.2	Threshold Predictor	0.9649	0.2100	0.2648
		Enhancer	0.9647	0.3213	0.3854
KL-subset	0.8	Threshold Predictor	0.9568	0.2936	0.3549
		Enhancer	0.9577	0.3586	0.4210
	0.9	Threshold Predictor	0.9533	0.2843	0.3369
		Enhancer	0.9531	0.3290	0.3820
	1.0	Threshold Predictor	0.9570	0.2559	0.3152
		Enhancer	0.9571	0.2971	0.3591
	1.1	Threshold Predictor	0.9615	0.2279	0.2949
		Enhancer	0.9614	0.2743	0.3459
	1.2	Threshold Predictor	0.9604	0.2199	0.2828
		Enhancer	0.9599	0.2516	0.3175

Scores of local neighborhoods

They are evaluated by means of log-odd ratios of neighboring residue pair frequencies in interfaces as opposed to residue pair frequencies on complementary protein surface areas. The resulting scores are averaged both over the neighborhood of the positions under study and the rows of the MSA associated with the protein.

On the basis of Figure 5 we enhanced *PresCont* for thresholds $\theta \in [0.500, 0.625]$. The decisive factor for this choice is that the *PresCont* score distributions for interface sites as well as non-interface positions above θ are "sufficiently close to" unimodal distributions. For every such θ, we set all scores less than or equal to θ to zero and then left the classification of all surface residues to the pCRF modified as follows. The residues of score zero are not taken into account when it comes to discretizing the protein characteristics (see Equations 4 and 5). Let us call this *enhancing above* θ.

Table 3 Comparing the enhancer with the threshold classifier of approximately equal specificity on synthetic scores assigned to surface residues of protein complexes taken from the CGNK-subset

Classifier	Specificity	Sensitivity	MCC
Threshold predictor	0.9399	0.3782	0.3387
Enhancer	0.9400	0.3104	0.2767

To evaluate improvements we proceeded as when compiling Table 2. For every threshold θ under consideration another threshold θ' was chosen such that thresholding at θ' has the same specificity as enhancing above θ. The results are displayed in Table 4 and visualized for an individual protein by Figure 6. According to Table 4 the increase in sensitivity ranges from 4% to 7%. The true-positive predictions on the surface of the protein with PDB-Entry 1QM4 are compared in Figure 6, where again the specificity of the two classifiers is the same.

Discussing posterior decoding

As in the case of linear-chain CRFs, the generalized Viterbi algorithm can be transformed into a variant form of the forward algorithm. It might be the case that the following additional problem arises.

Let $v_1, v_2, \ldots, v_n$ be the ordering in which the positions of $\mathcal{G}$ are traversed by the algorithm, and let $\widehat{I}$ denote the set of position indices $i < n$ such that v_i is not an element of the boundary $\mathcal{B}^{(i)}$ of the history set $\mathcal{H}^{(i)}$ at stage i. If $\widehat{I}$ is not empty, we encounter an obstacle when it comes to sampling label sequences. For $i \in \widehat{I}$, position v_i is not labeled in the course of the sampling procedure. That is why we augment the neighborhood graph $\mathcal{G}$ so that those positions no longer exist, all predictions remain unchanged, and the order of magnitude of the running time is not increased. To this end, we complement the

Table 4 Enhancing above various thresholds on *PlaneDimers*, where *PresCont*'s threshold was chosen such that the specificity approximately equals that of enhancing

	tp	tn	fp	fn	Spec.	Sen.	MCC
Enhancing above 0.500	2181	23182	4145	1414	0.848	0.607	0.362
PresCont	2100	23197	4130	1495	0.849	0.584	0.346
Enhancing above 0.525	2303	22917	4410	1292	0.839	0.641	0.373
PresCont	2206	22912	4415	1389	0.838	0.614	0.353
Enhancing above 0.550	2507	22103	5224	1088	0.809	0.697	0.375
PresCont	2419	22102	5225	1176	0.809	0.673	0.358
Enhancing above 0.575	2560	21992	5335	1035	0.805	0.712	0.380
PresCont	2463	21915	5412	1132	0.802	0.685	0.358
Enhancing above 0.600	2379	22685	4642	1216	0.830	0.662	0.376
PresCont	2253	22780	4547	1342	0.834	0.627	0.356
Enhancing above 0.625	2287	23044	4283	1308	0.843	0.636	0.376
PresCont	2136	23049	4278	1459	0.843	0.594	0.346

The sensitivity increased that way by 4%-7%. For every pair of experiments, the number of true negatives (tn), false negatives (fn), false positives (fp) and true positives (tp) are displayed.

ordering $v_1, v_2, \ldots, v_n$ as follows. For every $i \in \widehat{I}$, we insert a new node $\widehat{v}_i$ between v_i and v_{i+1}. Having extended the neighborhood graph by these nodes not being associated with residue positions of the protein under study and by new edges $\{v_i, \widehat{v}_i\}$ $(i \in \widehat{I})$, where for $i \in \widehat{I}$ and $y_0.y_1, y_2 \in \{\mathrm{N}, \mathrm{I}\}\Phi_{\widehat{v}_i}(y_0, \mathbf{x}) = \Phi_{\{v_i,\widehat{v}_i\}}(y_1, y_2, \mathbf{x}) = 0$, the above mentioned obstacle is eliminated without any influence on the prediction and the order of magnitude of the running time.

Proceeding now in a way analogous to the classical case, in every formula that is a building block of the generalized Viterbi algorithm the following two steps of replacement need to be performed.

First, for every position $i \in \mathcal{V}$, every edge $(i, j) \in \mathcal{E}$, every label $y_0 \in \{\mathrm{I}, \mathrm{N}\}$, and every label pair $(y_1, y_2) \in \{\mathrm{I}, \mathrm{N}\}^2$, we replace $\Phi_i(y_0, x)$ with $\exp(\Phi_i(y_0, x))$, and $\Phi_{\{i,j\}}(y_1, y_2, x)$ with $\exp\left(\Phi_{\{i,j\}}(y_1, y_2, x)\right)$.

Second, we replace sums with products and then maxima with sums.

Thus we obtain as analogues of the Viterbi variables $\mathrm{vit}_{\mathcal{H}_\mu}(\mathbf{y}_{\mathcal{B}_\mu})$ defined by Equation 6 what we call component forward variables $\mathrm{cf}_{\mathcal{H}_\mu}(\mathbf{y}_{\mathcal{B}_\mu})$.

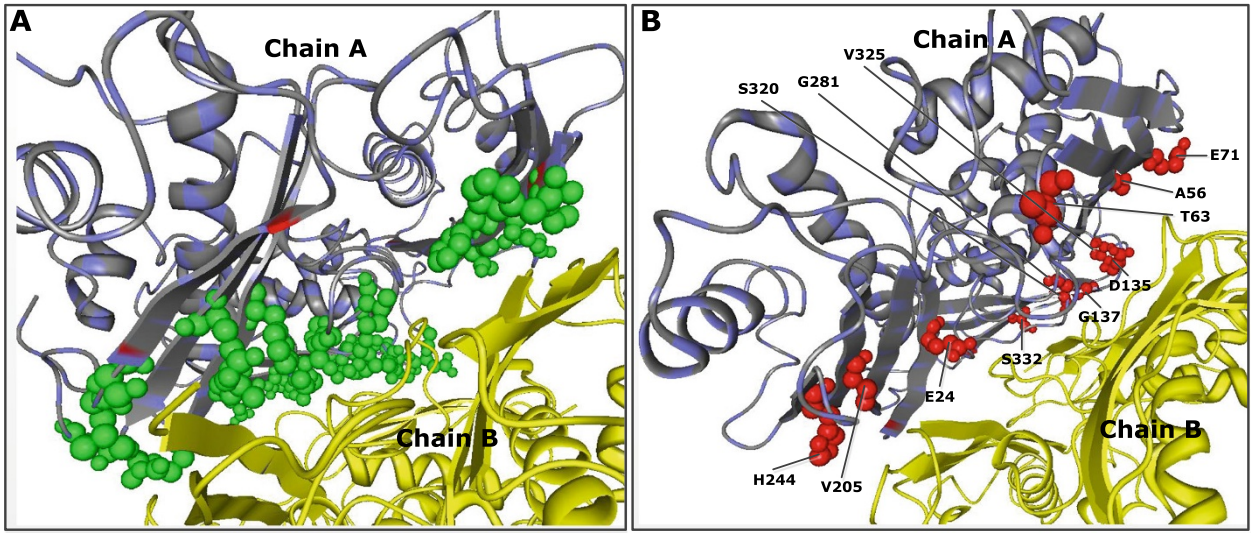

Figure 6 Comparison of enhancer and *PresCont* service of same specificity on the protein with PDB-Entry 1QM4. (A) Green spheres on the left show the interface surface residues correctly predicted by both tools. **(B)** Red spheres on the right indicate *additional* true positives of the enhancer.

If $\mathcal{H}_1^{(i)}, \mathcal{H}_2^{(i)}, \ldots, \mathcal{H}_{m_i}^{(i)}$ and $\mathcal{B}_1^{(i)}, \mathcal{B}_2^{(i)}, \ldots, \mathcal{B}_{m_i}^{(i)}$ are the connected components of the history set $\mathcal{H}^{(i)}$ and the corresponding boundary set $\mathcal{B}^{(i)}$ at stage $i \in \{1, 2, \ldots, n\}$, respectively, then the forward variable at stage i with respect to a boundary assignment $\mathbf{y}_{\mathcal{B}^{(i)}}$ is defined as

$$\mathrm{f}_i\left(\mathbf{y}_{\mathcal{B}^{(i)}}\right) := \prod_{j=1}^{m_i} \mathrm{cf}_{\mathcal{H}_j^{(i)}}\left(\mathbf{y}_{\mathcal{B}_j^{(i)}}\right).$$

For any assignment $\mathbf{y}_{\mathcal{B}^{(i)}}$ $(i > 1)$, the forward variable $\mathrm{f}_i(\mathbf{y}_{\mathcal{B}^{(i)}})$ is a nontrivial linear combination of forward variables $\mathrm{f}_{i-1}(\mathbf{y}_{\mathcal{B}^{(i-1)}})$, where $\mathbf{y}_{\mathcal{B}^{(i-1)}}$ ranges over some assignments of the boundary set $\mathcal{B}^{(i-1)}$ at stage $i-1$. Analogous to the linear-chain case, a random backward walk through a state graph, with all possible assignments $\mathbf{y}_{\mathcal{B}^{(i)}}$ $(i = n, n-1, \ldots, 1)$ being the set of nodes, results in a random labeling of the positions, where each labeling is drawn with its posterior probability.

This sampling technique allows the efficient calculation of posterior probabilities at nodes and edges in a straightforward manner.

Conclusions

Residue-wise score-based threshold predictors of protein-protein interaction sites assign to each residue of the protein under study a score. The classification is then made by thresholding the score. In case of using probabilistic data models, the parameters of the threshold predictor have been learned on a training data set in advance.

We have demonstrated that such threshold predictors can be improved by pCRF-based enhancers given the shape of the interface surface score distribution and the non-interface surface score distribution with respect to the training set resemble the shape of unimodal distributions. Besides the surface residue scores, only the spatial neighborhood structure between the surface residues of the protein under study is taken into account. Thus, the improvement can be attributed to our model. In addition to the precision of the scores, the amount of improvement depends on the 3D-complexity of the interfaces to be predicted. To this end, three sets of experiments with synthetic surface residue scores for protein complexes randomly chosen from the data set *PlaneDimers* compiled by Zellner *et al.* [25] and from the lists published by Keskin *et al.* [39] and Cukuroglu *et al.* [40].

The enhancement is structurally based on the following model property of pCRFs in contrast to residue-wise predictors. Though the scores of near-by residues may be correlated, labeling a position as interface or non-interface by thresholding the score does *not* influence the classification of its neighbors. When using pCRFs, this is the case.

The pCRF-based enhancer is also applicable, if the score distributions are only unimodal over a certain sub-domain. The improvement is then restricted to that domain. Thus we were able to improve the prediction of the *PresCont* server devised by Zellner *et al.* on *PlaneDimers* [25].

The prediction is made on grounds of a generalized Viterbi inference heuristic. As for training, we developed a piecewise training procedure for pCRFs, where the enhancer needs to be trained on data originating from the same source as the training data of the threshold predictor to be improved.

A prototypical implementation of our pCRF-based method is accessible at http://ppicrf.informatik.uni-goettingen.de/index.html.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

The model was designed by ZD and KW, who also adapted piecewise training to pCRFs and performed the analysis of the first submission. The tool was designed and implemented by ZD and KW, who were supported by MW and TW. The algorithms were devised by MG, MS, MW and SW. The generalized Viterbi algorithm, however, was conceived by MS. The Web server was set up by MG. The analysis necessary for the revised version was performed by TKLD, who was supported by MW. The manuscript was drafted by ZD and KW, written by MS and SW and revised by TW. SW conceived the study. All authors read and approved the final manuscript.

Acknowledgements

Zhijie Dong acknowledges financial support by the Deutsche Forschungsgemeinschaft (Research Training Group 1023 "Identification in Mathematical Models: Synergy of Stochastic and Numerical Methods"). Moreover, we thank Rainer Merkl and Hermann Zellner from Regensburg, who supported us in gaining a deeper understanding of the *PresCont* service. Special thank goes to Moritz Manecke for a first implementation of the generalized Viterbi algorithm. Finally, we acknowledge with thanks the valuable suggestions and comments of the unknown referees.

Author details

[1]Institute of Computer Science, University of Göttingen, Goldschmidtstr. 7, 37077 Göttingen, Germany. [2]Institute of Bioinformatics, University of Göttingen, Goldschmidtstr. 1, 37077 Göttingen, Germany. [3]Institut für Mathematik und Informatik, Walther-Rathenau-Str. 47, 17487 Greifswald, Germany.

Received: 11 October 2013 Accepted: 1 August 2014
Published: 13 August 2014

References

1. Sowa ME, He W, Slep KC, Kercher MA, Lichtarge O, Wensel TG: **Prediction and confirmation of a site critical for effector regulation of RGS domain activity.** *Nat Struct Biol* 2001, **8:**234–237.
2. Zhou HX: **Improving the understanding of human genetic diseases through predictions of protein structures and protein-protein interaction sites.** *Curr Med Chem* 2004, **11:**539–549.
3. Zhou HX, Qin S: **Interaction-site prediction for protein complexes: a critical assessment.** *Bioinformatics* 2007, **23**(17):2203–2209.
4. Li JJ, Huang DS, Wang B, Chen P: **Identifying protein-protein interfacial residues in heterocomplexes using residue conservation scores.** *Int J Biol Macromol* 2006, **38**(3–5):241–247.
5. Kufareva I, Budagyan L, Raush E, Totrov M, Abagyan R: **PIER: protein interface recognition for structural proteomics.** *Proteins* 2007, **67**(2):400–417.
6. Burgoyne NJ, Jackson RM: **Predicting protein interaction sites: binding hot-spots in protein-protein and protein-ligand interfaces.** *Bioinformatics* 2006, **22**(11):1335–1342.

7. de Vries SJ, van Dijk AD, Bonvin AM: **WHISCY: what information does surface conservation yield? Application to data driven docking.** *Proteins* 2006, **63**(3):479–489.
8. Hoskins J, Lovell S, Blundell TL: **An algorithm for predicting protein-protein interaction sites: abnormally exposed amino acid residues and secondary structure elements.** *Protein Sci* 2006, **15**(5):1017–1029.
9. Landau M, Mayrose I, Rosenberg Y, Glaser F, Martz E, Pupko T, Ben-Tal N: **ConSurf2005: the projection of evolutionary conservation scores of residues on protein structures.** *Nucleic Acids Res* 2005, **33**(Web-Server-Issue):299–302.
10. Liang SL, Zhang C, Liu S, Zhou Y: **Protein binding site prediction using an empirical scoring function.** *Nucleic Acids Res* 2006, **34**(13):3698–3707.
11. Murakami Y, Jones S: **SHARP^2: protein-protein interaction predictions using patch analysis.** *Bioinformatics* 2006, **22**(14):1794–1795.
12. Zhou HX, Shan Y: **Prediction of protein interaction sites from sequence profile and residue neighbor list.** *Protein Struct Funct Genet* 2001, **44:**336–243.
13. Fariselli P, Pazos F, Valencia A, Casadio R: **Prediction of protein-protein interaction sites in heterocomplexes with neural networks.** *Eur J Biochem* 2002, **269:**.
14. Ofran Y, Rost B: **Predicted protein-protein interaction sites from local sequence information.** *FEBS Lett* 2003, **544:**236–239.
15. Chen H, Zhou HX: **Prediction of interface residues in protein-protein complexes by a consensus neural network: test against NMR data.** *Protein Struct Funct Genet* 2005, **61:**21–35.
16. Ofran Y, Rost B: **ISIS: interaction sites identified from sequence.** *Bioinformatics* 2007, **23**(2):13–16.
17. Porollo A, Meller J: **Prediction-based fingerprints of protein-protein interactions.** *Protein Struct Funct Genet* 2007, **66:**630–645.
18. Bordner A, Abagyan R: **Statistical analysis and prediction of protein-protein interfaces.** *Protein Struct Funct Genet* 2005, **60:**353–366.
19. Bradford J, Westhead D: **Improved prediction of protein-protein binding sites using a support vector machines approach.** *Bioinformatics* 2005, **21**(8):1487–1494.
20. Chung JL, Wang W, Bourne PE: **Exploiting sequence and structure homologs to identify protein-protein binding sites.** *Proteins* 2006, **62:**630–640.
21. Koike A, Takagi T: **Prediction of protein-protein interaction sites using support vector machines.** *Protein Eng Design Selec* 2004, **17**(2):165–173.
22. Res I, Mihalek I, Lichtarge O: **An evolution-based classifier for prediction of protein interfaces without using protein structures.** *Bioinformatics* 2005, **21**(10):2496–2501.
23. Wang B, Wong HS, Huang DS: **Inferring protein-protein interaction sites using residue conservation and evolutionary information.** *Protein Pept Lett* 2006, **13**(10):999–1005.
24. Wang B, Chen P, Huang DS, Li JJ, Lok TM, Lyu MR: **Predicting protein interaction sites from residue spatial sequence profile and evolution rate.** *FEBS Lett* 2006, **580**(2):380–384.
25. Zellner H, Staudigel M, Trenner M, Bittkowski M, Wolowski V, Icking M, Merkl R: **PresCont: Predicting Protein-Protein Interfaces Utilizing Four Residue Properties.** *Proteins: Struct Funct Bioinformatics* 2011, **80**(1):154–168.
26. Neuvirth H, Raz R, Schreiber G: **ProMate: a structure based prediction program to identify the location of protein-protein binding sites.** *J Mol Biol* 2004, **338**(1):181–199.
27. Bradford JR, Needham CJ, Bulpitt AJ, Westhead DR: **Insights into protein-protein interfaces using a Bayesian network prediction method.** *J Mol Biol* 2006, **362**(2):365–386.
28. Li MH, Lin L, Wang XL, Liu T: **Protein-protein interaction site prediction based on conditional random fields.** *Bioinformatics* 2007, **23**(5):597–604.
29. Hwang H, Vreven T, Weng Z: **Binding interface prediction by combining protein-protein docking results.** *Proteins: Struct Funct Bioinformatics* 2013. [http://dx.doi.org/10.1002/prot.24354]
30. Lafferty JD, McCallum A, Pereira FCN: **Conditional random fields: probabilistic models for segmenting and labeling sequence data.** In *Proceedings of the Eighteenth International Conference on Machine Learning*. Edited by Brodley CE. San Francisco, CA, USA: Danyluk AP. Morgan Kaufmann Publishers Inc.; 2001:282–289. [http://dl.acm.org/citation.cfm?id=645530.655813]
31. Sutton C, McCallum A: *Introduction to Statistical Relational Learning*. Cambridge, Massachusetts, USA: MIT Press; 2006 chap. An Introduction to Conditional Random Fields for Relational Learning.
32. McCallum A, Li W: **Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons.** In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4,* CONLL '03. Stroudsburg: Association for Computational Linguistics; 2003:188–191. [http://dx.doi.org/10.3115/1119176.1119206]
33. Dietterich TG, Ashenfelter A, Bulatov Y: **Training conditional random fields via gradient tree boosting.** In *Proceedings of the Twenty-first International Conference on Machine Learning, Volume 69 of ACM International Conference Proceeding Series*. Edited by Brodley CE. New York, NY, USA: ACM; 2004:28. [http://doi.acm.org/10.1145/1015330.1015428]
34. Zhu J, Nie Z, Wen JR, Zhang B, Ma WY: **2D Conditional, Random Fields for Web information extraction.** In *Proceedings of the 22Nd International Conference on Machine Learning, Volume 119 of ACM International Conference Proceeding Series*. Edited by Raedt LD, Wrobel S. New York, NY, USA: ACM; 2005:1044–1051. [http://doi.acm.org/10.1145/1102351.1102483]
35. Sutton C, McCallum A: **Piecewise training of undirected models.** In *UAI '05, Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI), Edinburgh, Scotland, July 26-29.* AUAI Press; 2005:568–575.
36. McCallum A, Rohanimanesh K, Sutton C: **Dynamic conditional random fields for jointly labeling multiple sequences.** In *NIPS-2003 Workshop on Syntax, Semantics and Statistics.* 2003.
37. Sha F, Pereira F: **Shallow parsing with conditional random fields.** 2003. [citeseer.ist.psu.edu/article/sha03shallow.html]
38. Liu DC, Nocedal J: **On the limited memory BFGS method for large scale optimization.** *Math Program* 1989, **45:**503–528.
39. Keskin O, Tsai CJ, Wolfson H, Nussinov R: **A new, structurally nonredundant, diverse data set of protein-protein interfaces and its implications.** *Protein Sci* 2004, **13:**1043–1055.
40. Cukuroglu E, Gursoy A, Nussinov R, Keskin O: **Non-redundant unique interface structures as templates for modeling protein interactions.** *PLoS ONE* 2014, **9:**e86738.
41. Rost B, Sander C: **Conservation and prediction of solvent accessibility in protein families.** *Proteins: Struct Funct Bioinformatics* 1994, **20**(3):216–226. [http://dx.doi.org/10.1002/prot.340200303]
42. Hildebrandt A, Dehof AK, Rurainski A, Bertsch A, Schumann M, Toussaint NC, Moll A, Stöckel D, Nickels S, Mueller SC, Lenhof HP, Kohlbacher O: **BALL - biochemical algorithms library 1.3.** *BMC Bioinformatics* 2010, **11:**531.
43. Xia JF, Zhao XM, Song J, Huang DS: **APIS: accurate prediction of hot spots in protein interfaces by combining protrusion index with solvent accessibility.** *BMC Bioinformatics* 2010, **11:**174.
44. Miller S, Janin J, Lesk AM, Chothia1 C: **Interior and surface of monomeric proteins.** *J Mol Biol* 1987, **196**(3):641–656.
45. Larsen TA, Olson AJ, Goodsell DS: **Morphology of protein–protein interfaces.** *Structure* 1998, **6**(4):421–427.
46. Bouvier B, Grünberg R, Nilges M, Cazals F: **Shelling the Voronoi interface of protein–protein complexes reveals patterns of residue conservation, dynamics, and composition.** *Proteins: Struct Funct Bioinformatics* 2009, **76**(3):677–692.

doi:10.1186/1471-2105-15-277
Cite this article as: Dong *et al.*: CRF-based models of protein surfaces improve protein-protein interaction site predictions. *BMC Bioinformatics* 2014 **15**:277.