

Fast Screening Algorithms for Protein Interface Structural Alignments

by

Ali Tugrul Balci

A Dissertation Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Computer Science



July 31, 2018

Fast Screening Algorithms for Protein Interface Structural Alignments

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Ali Tugrul Balci

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

Prof. Attila Gürsoy (advisor)

Prof. Özlem Keskin Özkaya (co-advisor)

Prof. Alper Erdoğan

Assoc. Prof. Arzucan Özgür

Asst. Prof. Barış Akgün

Date: _____



Pain is weakness leaving the body

ABSTRACT

Protein-protein interactions (PPIs) form the basis of many biological processes in living organisms. The significance of PPIs in mediating biological activity necessitates the identification of novel interactions. Template based structural alignment is one of the computational approaches to predict protein-protein interactions using known protein interfaces. One challenge in template-based prediction is the computational cost due to the one-to-all comparison of the query protein against a database of all known interfaces. In this thesis, two different approaches have been developed a) QuickRet, a hashing based algorithm, b) and a deep learning based algorithm. QuickRet, a fast screening algorithm, ranks interfaces due to their structural similarity to a query protein. It extracts features (angles and distances derived from four atoms) from structures of interfaces and compares them with the features extracted from the query protein. QuickRet is tested with the PIFACE database, a clustered protein-protein interface database, and predictions made by the template interface based PPI prediction algorithm, PRISM. The results indicate that QuickRet is successful in filtering structurally dissimilar interfaces for a given protein. With at least 80% match, 99% (320/43500 interface structures remained) of the database is eliminated and the average RMSD value of the remaining structures is 2.4Å. With at least 90% match, 99.9% (50/43500 structures remained) of the database is eliminated and the average RMSD value drops to 2.28Å. In addition, a deep learning based method which predicts, for a given protein complex, if the interface between the proteins of the complex is a true interface or not (based on known interfaces in Protein Data Bank). The model, a 3-dimensional convolutional model, analyzes the given structure and outputs the probability of the given structure being an interface. The accuracy of the model for several interface data sets, including PIFACE, PPI4DOCK,

DOCKGROUND is approximately 80%. Both algorithms can be used to reduce the computational cost of template based PPI predictions.



ÖZETÇE

Birçok biyolojik işlemin temelini protein-protein etkileşimleri (PPE) oluşturur. PPElerin biyolojik aktivitelerin yerine getirilmesindeki önemi yeni PPElerin tanımlanmasını önemli kılar. Hataya karşı dayanıklı ve güvenilir bilimsel araçlar yeni etkileşimlerin bulunmasını kolaylaştırmak için kullanılabilir. Bilgisayar üzerinden yeni etkileşimlerin bulunmasında en büyük zorluklardan biri arayüzlerden oluşan bir veri kümesinde verilen protein yapısına en uygun arayüzü bulmaktır. Bu tezde, bir veri kümesindeki arayüzleri verilen proteine olan yapısal benzerliklerine göre sıralayan QuickRet algoritmasını sunuyorum. Algoritma arayüzlerden çıkartılan yapısal karakteristik özellikleri verilen proteinden çıkartılan yapısal karakteristik özelliklerle karşılaştırır. QuickRet'i yapısal özelliklerine göre kümelenmiş arayüzlerden oluşan veritabanı PIFACE'si ve örnek arayüzleri kullanarak PPE tahmini yapan algoritma PRISM'in tahminlerini kullanarak test ettim. Sonuçlar QuickRet'in verilen proteinler için yapısal olarak benzer arayüzler bulmakta başarılı olduğunu gösteriyor. PIFACE veri kümesinde, QuickRet verilen 166 kümesi üyesinden 56'si için küme temsilcilerini 1. sırada veriyor. PRISM tahminleri içinse, verilen 80 proteinden 56'si için PRISM tarafından tahmin edilen arayüz yapısını %50'lik dilimin içerisinde sıralıyor. QuickRet'e ek olarak, arayüz bölgesini tahmin eden bir derin öğrenme modeli sunuyorum. Model, 3 boyutlu konvolüsyon ağı, verilen yapıyı analiz eder ve verilen yapının arayüz olma olasılığını hesaplar. Model PIFACE, PPI4DOCK, DOCKGROUND'un içinde bulunduğu veri kümelerindeki arayüz yapılarını %80 doğruluk ile tahmin etti.

ACKNOWLEDGMENTS

First things first, I would like to thank my advisors Prof. Attila Gürsoy and Prof. Özlem Keskin Özkaya for their advise and support. I would also like to thank the committee members for their time and suggestions.

I would like to thank my lab members. Befriending new people is always difficult for me, but they made it too easy.

I would like to thank my high school friends for always being there for me. It is hard to call them just friends, they are my brothers and sisters.

Last but not least, I would like to thank my family for the endless love they provide.

TABLE OF CONTENTS

List of Tables	xi
List of Figures	xii
Nomenclature	xiv
Chapter 1: Introduction	1
Chapter 2: Literature Survey	3
2.1 Alignment	3
2.2 Machine Learning	9
2.3 Scoring Functions	13
2.4 Data	17
Chapter 3: Methods	23
3.1 QuickRet	23
3.1.1 Feature Set and Base Set	24
3.1.2 Creating Hash Table	25
3.1.3 Retrieval	26
3.1.4 Scoring	29
3.1.5 Transformation Parameters	29
3.1.6 Data	33
3.1.7 Complexity	33
3.2 Interface Prediction with Deep Learning Methods	34
3.3 Models	34

3.4	Weight Initialization and Training	36
3.5	Data	36
Chapter 4:	Results and Discussion	41
4.1	QuickRet	41
4.1.1	Test 1: Top Matching Interfaces	42
4.1.2	Test 2: TMalign Assessments of QuickRet's results	48
4.1.3	Test 3: PIFACE Members	51
4.1.4	Test 4: PRISM Predictions	53
4.1.5	Test 5: Algorithm Speed	57
4.2	Interface Prediction with Deep Learning Methods	61
Chapter 5:	Conclusion	65
	Appendices	72
Appendix A:	APPENDIX A	73

LIST OF TABLES

3.1	Distribution of negative and positive samples.	39
3.2	Alternative distribution of negative and positive samples.	40
4.1	Alignment of top 1 interface structures with the given proteins. CM: Correct matches IL: Interface length in residues PM: percentage of interface matched with the protein PL: Protein length in residues . .	42
4.2	Matching pairs in alignment of 1n2dAC_C - 1folA	47
4.3	Matching pairs in alignment of 1n1iAD_D - 3aqjA	47
4.4	Rankings of QuickRet and TMaling for 1folA.	50
4.5	Execution time of TMalign and QuickRet.	60
4.6	Results of the second model for the two datasets.	62
A.1	Filtering interface structures according to percentage match. #int: Number of interface structures which passes the filter M_{RMSD} : Average RMSD σ_{RMSD} : standard deviation for RMSD	73
A.2	Filtering interface structures according to percentage contact match. #int: Number of interface structures which passes the filter M_{RMSD} : Average RMSD σ_{RMSD} : standard deviation for RMSD	82

LIST OF FIGURES

2.1	Formula of Amino Acids [30].	3
2.2	Four stages of a protein structure[28].	4
3.1	The features and the base.	25
3.2	Architecture of the first model.	35
3.3	Architecture of the second model.	37
4.1	Histogram of percentage of correctly matched residues.	46
4.2	Histogram of RMSD.	46
4.3	Alignment of 1n2dAC_C with 1folA.	46
4.4	Alignment of 1n1iAD_D with 3aqjA.	46
4.5	Histogram of RSMD calculated by TMalign for the top ranked structures.	49
4.6	Histogram of the representatives' ranks according to QuickRet.	52
4.7	Histogram of the representatives' ranks according to the alternative scoring function.	52
4.8	Alignment of 1qqjAB_A with 2hzyA.	53
4.9	Alignment of 2wojAD_A with 2hzyA.	53
4.10	Structure of 1qqjAB_A.	53
4.11	Structure of 2wojAD_A.	53
4.12	Graph of the average number of remaining interface structures for given percentage match thresholds.	55
4.13	Graph of the average RMSD value of remaining interface structures with the given protein for given percentage match thresholds.	55

4.14	Graph of the average number of remaining interface structures for given percentage contact match thresholds.	55
4.15	Graph of the average RMSD value of remaining interface structures with the given protein for given percentage contact match thresholds.	55
4.16	Histogram of the templates' ranks according to QuickRet.	56
4.17	Histogram of the templates' ranks according to the alternative scoring function.	56
4.18	Alignment of 1tsrAB_B with 1tupB.	57
4.19	Structure of 1tsrAB_B.	57
4.20	Alignment of 1nvuRS_R with 1nvwR.	58
4.21	Alignment of 2f4iAB_B with 1nvwR.	58
4.22	Structure of 1nvuRS_R.	58
4.23	Structure of 2f4iAB_B.	58
4.24	Graph of the length of the longest list in the hash table vs thresholds.	59
4.25	Graph of the number of bins in the hash table vs thresholds.	59
4.26	Graph of the execution time vs thresholds.	59
4.27	Distributions of acidic composition percentages.	63
4.28	Distributions of aliphatic composition percentages.	63
4.29	Distributions of aromatic composition percentages.	63
4.30	Distributions of basic composition percentages.	63
4.31	Distributions of hydroxyl composition percentages.	64
4.32	Distributions of imino composition percentages.	64
4.33	Distributions of sulfur composition percentages.	64

NOMENCLATURE

PDB Protein Data Bank

PRISM Protein Interactions by Structural Matching

ReLU Rectified Linear Unit

Chapter 1

INTRODUCTION

The interactions between proteins are one of the basic constituents of the life. Understanding these interactions between proteins leads to many breakthroughs in the fields of biology, genetics, medicine and so on. Importance of understanding protein protein interactions makes predicting interactions between two proteins and predicting the structure of interactions some of the most important challenges in the computational biology. Importance of discovering new interactions makes two tasks very important. First, to be able to filter a database of interfaces and eliminate interface structures which are not similar to the given protein structure accelerate the process of discovering new interactions. Second, by being able to predict interface structures, we can predict the possibilities of hypothesized interfaces. In this thesis, we worked on two different projects which are related to the protein - protein interactions. First of them is a filtering/retrieval tool which uses geometric hashing. Second of them is a deep learning algorithm which predicts the interface regions.

In the first project, we created a mechanism, QuickRet, to filter a dataset of interfaces for the given protein structure. The filtering is done by structural alignment. The structural alignment is the process of finding similar 3 dimensional patterns in the given interfaces and then minimizing the distance between the similar structures. Since the functions of the proteins mainly depend on the structure of the proteins rather than the sequence of the proteins, filtering based on structural alignment is more robust. To filter a database of the interfaces fast, we created one hash table containing features from all interfaces in the database. While QuickRet is very fast, it also retrieves similar structures to the query proteins.

The second project we worked on is a 3 dimensional deep neural network to predict the interface structure of the interacting complex. We converted structures into 3 dimensional arrays and trained convolutional networks on these data. We showed that the model is successful at identifying the correct interface structure with an accuracy of about 80%.

The rest of the thesis is organized in this way: The next chapter surveys the literature for the tools and technique we have used in our project or we have tried to improve. In the chapter 3, we describe our algorithm QuickRet and the deep learning model. Also we explained the data we used and preprocessing we applied on them if there are any. In the chapter 4, we presented our results and analyzed these results to find strengths and weaknesses of our tools. Finally, in the chapter 5, conclusion, we shared overviews of the algorithms and suggested future improvements.

Chapter 2

LITERATURE SURVEY

2.1 Alignment

The proteins go through a transformation after they are synthesized. At first, they are a single thread of amino acids, the single thread is called the primary structures of proteins. The chemical formula for amino acids are given in the Figure 2.1. In the Figure 2.1, the carbon atom to which every group bounds to is called C_α (carbon alpha). There are 20 common amino acid types which are frequently seen in the protein structures. In the literature, the name of the amino acids are abbreviated in 3 letters or 1 letter. The amino acids bond to each other by a bond named "peptide bond". The peptide bond is formed between the carboxyl group and the amino group. The carboxyl group and the amino group are shown in the Figure 2.1.

After a protein is synthesized, the single thread begins to fold due to the chemical properties of the amino acids. These folding process gives a shape to proteins. Structures of proteins have repeating patterns which are called secondary structures. Secondary structures are very stable. Hydrogen bonds are the main force behind the

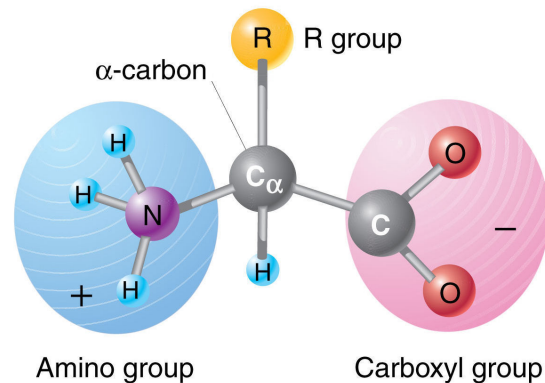


Figure 2.1: Formula of Amino Acids [30].

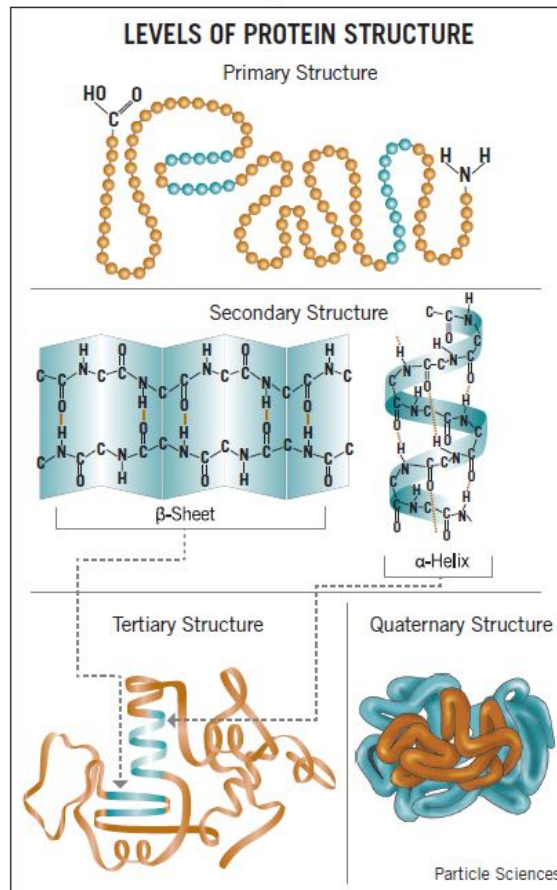


Figure 2.2: Four stages of a protein structure[28].

formation of secondary structures. Hydrogen bonds are partially electrostatic attractions which occur between the oxygen of the carboxyl groups and the hydrogen of the amino groups. There are two types of secondary structures; helices, and beta sheets which are shown in Figure 2.2.

After enough time, the thread takes a 3 dimensional shape which is called tertiary structure. The tertiary structure is the most stable form the protein can take. Aside from the hydrogen bonds, and the peptide bond, there are interactions between the side chains (R groups in the Figure 2.1) which makes the tertiary structure the most stable form for a protein. After a protein takes its tertiary structure, it interacts with other proteins to form complexes which are called quaternary structure. These steps are all shown in the Figure 2.2.

The proteins perform their tasks by physically interacting with other proteins or substances. This deems the structure of the proteins very important research topic. To understand how proteins perform a wide variety of different and very important tasks, we should analyze the structure. To analyze the structure, scientists model protein structures in computers with techniques such as X-ray crystallography and nuclear magnetic resonance (NMR). In protein structure files, scientists keep information regarding the structure such as three dimensional positions for the atoms, secondary structures, subunits, and so on. From the information on protein structures, higher level information can be derived. For example, bond length, bond angles and torsion angles are higher level information. Bond length is the distance between two bonded atoms for example distance between N and C gives the length of peptide bond. The bond angle is the angle between two consecutive bonds. Finally, the torsion angle is the angle between two planes defined by 4 consecutive atoms in the backbone of the protein structure. Backbone is the structure defined by the C_α atoms. These informations are illustrated in Figure 3.1. Transferring structure information to computers leads to more robust and rigorous analysis. One example of analysis is alignment.

To find similarities between proteins, scientists align them. Alignment means finding similarities and presenting them side by side. There are three types of alignments; sequence alignment, structural alignment and hybrid of both. While the amino acid sequence of a protein gives the protein its structure, aligning sequences is usually ineffective. Small changes in a sequence may lead to a drastic change in the structure of the protein. Also, sequences which seem very different may form similar structures which indicate that the proteins perform similar functions. While structural alignment is more robust than sequence alignment, the hybrid approaches are also very powerful. By using every available information, the hybrid approaches produces adequate alignments. In our research, we used a method similar to the geometric hashing. Our method is a structural alignment tool.

One application of the structural alignment is to find if an interface and a protein

structure are similar. Finding these similarities can lead us to the discovery of novel protein protein interactions. The idea is that if some parts of the surface of two proteins are similar to the sides of an interface, we can use the interface as a template to dock these interface to predict the interacting model between these proteins. This is the principle behind protein protein interaction prediction tools such as PRISM[1] and SnapDock[10]. These new interaction models can help scientists to discover new novel interactions between proteins which are not known to interact previously. Finding novel protein protein interactions are important for understanding functions and roles of the proteins. In addition to that, since proteins have a role in nearly every biological event, finding new interactions reveals information about how the biological events happen such as diseases, cell division and many more. One example is that there are proteins which have roles in the control of the cell division. If a mutation occurs in the interface regions of these proteins, it may lead to cancer. Knowing the interfaces to which mutations happen may help scientists understand the effect of mutations in protein-protein signaling and help developing new drugs for cancer.

Nussinov and Wolfson described a structural alignment method with geometric hashing[27]. Before the alignment tools, to structurally align molecules, scientists used visualization programs. Human involvement was making the process error-prone and impossible to align large number of molecules. Therefore, Nussinov and Wolfson suggested adapting computer vision algorithms to structural alignment of molecules. In computer vision and graphics, there are algorithms applicable to point clouds which can be easily adapted to molecules. Molecules can be represented as point clouds; if molecules are considered to be sets of atoms which can be considered as points with their three dimensional coordinates in return. The algorithm they applied to structural alignment was geometric hashing [20].

Geometric hashing algorithm is originally applied in robotics. The task is to make robots recognize certain objects. To achieve this, Lamdan et al. used hash table data structure to retrieve models based on features extracted from the same models. They extracted rotation and translation invariant features from the images. After

the feature extraction, the image turns into a point cloud with a point located on every feature coordinate. Using 2 points among other points from the same model, in two dimensions, they constructed reference frames. They name these pairs from which a new reference frame is constructed reference sets. Having constructed the reference frame, they applied the change in the reference frame to other model points. The new quantized coordinates are key values for the hash table while models and reference sets are values. First phase of the geometric hashing algorithm, can also be called learning phase, is finished after above mentioned procedure is carried out for every model, and the hash table is created. In the second phase is the online phase of the algorithm. When the robot sees an object, it recognizes the object by retrieving candidate models using the features extracted from the newly scene object. Lamdan et al. developed a voting scheme for this task. The algorithm extracts features from the object as mentioned above. It retrieves models from the hash table using new quantized coordinates of the model points calculated according to the reference frames. The algorithm casts a vote for every retrieved model. If a model passes predetermined threshold then the algorithm returns the model as recognized model. If none of the model passes the threshold then the recognition phase continues with selection of another reference set.

Geometric hashing algorithm offers many advantages for the structural alignment of protein structures. The main advantage of the algorithm is robustness against missing information. In computer vision, the algorithm works fine even when some part of the object is occluded. The voting scheme, the algorithm operates, does not need to obtain the number of the votes equal to the number of entry belonging to the model in the hashtable. By thresholding votes, if the newly seen model is in the database, even if part of the model is obscured, the rest of the model would produce enough vote to pass the threshold. After the voting phase, using intrinsic properties of the models, a validation model can be designed as in the case of MultiProt. Another advantage of geometric hashing is its invariance to rotation and translation. Since the coordinates are recalculated for every reference frame and the reference sets are cho-

sen among the model points, the translation and the rotation of the model does not change anything in the recognition phase. Since every pdb file has its own reference frame and the patterns in the protein can be anywhere in any rotation, the invariance property of the geometric hashing constitutes an important advantage for the structural alignment. Final advantage of geometric hashing is the sequence independence. Because the primary structure of the proteins bends in the three dimensional space, a motif in the structure does not have to include consecutive amino acids in the primary structures of the protein. By modeling structures as point clouds, geometric hashing algorithm does not need any information on sequence of the protein. Thus, geometric hashing eliminates the need for the parameters such as gap, insertion and deletion scores on which sequence based alignment tools heavily depend. Being sequence independent also is a major advantage for interface-protein alignment, since amino acids from anywhere in the sequence can comprise of the interface.

Nussinov and Wolfson adapted this algorithm for structural alignment with few changes. First, they chooses three noncollinear atoms with which they will construct a reference frame. These three atoms, a_1 , a_2 , a_3 defines a plane in 3 dimensional space. The reference frame is centered on the coordinates of the first atom, a_1 . The $a_2 - a_1$ direction defines the x-axis and the direction orthogonal to x-axis in counterclockwise defines the y-axis. Nussinov and Wolfson determined z-axis according to the right hand rule. After the reference frame is established, any point, v , in the space can be written as, assuming a_x , a_y , a_z as unit vectors, $v = a_x + a_y + a_z + a_0$. When a transformation is applied to the point, the variables does not change if the reference set is transformed with the same parameters. Speaking for a rigid model such as structure of the proteins, this holds true. Thus the coordinates according to the reference frame are invariant to the transformation. Having calculated the coordinates, they inserted model and reference set to a hash table using hash value calculated from the properly discretized coordinates and the atom labels. After the algorithm finishes recording every model in the hash table, the algorithm is ready for the matching phase.

In the matching phase, Nussinov and Wolfson applied above mentioned procedure

to the given protein to obtain hash values, and model and reference set pairs. With the hash values, the algorithm retrieves the model and reference set pairs from the hash table. If the number of votes for a specific pair exceeds a predefined threshold, the model and the reference set pass to the next validation step. In the validation step, the algorithm validates the alignment with biological informations such as interactions between the atoms. If the validation step fails, the algorithms switches back to voting phase. The algorithms continues until it finds a model or all reference sets are searched.

The algorithm which inspired our work in so many ways is SnapDock [10]. The algorithm is also a template based docking algorithm similar to PRISM. It filters a database of interface structures by using features extracted from structures. It quickly compares the signature from interface structures and the given proteins and eliminates the interface structures which are not similar to the given proteins. After finding the similar interface, it calculates alignment between the interfaces and the given proteins by using pose clustering method [33]. The algorithm aligns the given proteins to each side of the interface. Finally, the docked complex is refined and ranked.

To the best of our knowledge no work has been done on prediction of interfaces using machine learning methods. However, studies on machine learning and bioinformatics inspired this work. I will analyze them in the following section.

2.2 Machine Learning

Machine learning is a subbranch of artificial intelligence in computer science. In general, the machine learning methods use statistical approaches to make inferences from data. Usually, these inferences are drawn from data without explicit instructions. Machine learning algorithms proved very useful especially when there is an abundance of data. One example for a field which benefits from machine learning algorithms is biology. Machine learning technique constitutes most of the computational biology. In this section, we summarize important researches on machine learning and the

intersection of machine learning and computational biology.

LeNet is a convolutional neural network which is originally designed to recognize hand written numbers[21]. LeNet is one of the first convolutional neural networks which contributed to the fame of the convolutional networks. Before, these networks, scientists were very skeptical about the abilities of neural networks because of the problems such as the local minima, technical difficulties in training networks, and linearity.

I will first explain convolutional neural networks to better analyze LeNet. Convolutional networks are special kinds of neural networks which have 3 distinct advantages over multilayer perceptrons. The first advantage is the kernels. For most of the machine learning problems, scientists should preprocess the input. For example, centering input is very common in pattern recognition. Another example for preprocessing is feature extraction. Before neural networks, the usual pipeline of pattern recognition was first feature extraction and then training. Convolutional neural networks do not need feature extraction, thanks to kernels. Kernels are the group of parameters applied on every where on the input. This property enables convolutional neural networks to extract their own features.

The second advantage is weight sharing. In general, every unit of a neural network has more than one parameter for the input field. However, kernels greatly reduce the number of parameters by assigning a parameter to more than one input field. While this specialization bolsters the convolutional networks, it also guarantees that the convolutional networks have less parameters compared to an ordinary neural network with the same capacity. Therefore, convolutional neural networks occupy less space in the memory and take less time to train.

The third advantage is subsampling. The convolutional neural networks include middle layers called “pooling units”. These units summarize the input in one of the three ways: they either calculate average or find the maximum or the minimum value. In general, most of the input is redundant. Pooling units omit redundancy. By doing so, pooling units save computational time while not reducing accuracy of the model.

Another advantage of pooling units is the robustness against noise. By summarizing the data, pooling units help networks to eliminate the noise. Thus they increase the accuracy of networks. Finally, pooling layers trim the number of input fields by summarizing the input. Therefore they reduce the number of parameters and the memory requirement of networks.

Residual networks differ from other networks due to their depth[14]. The size of the stacked layers in a residual networks is usually 10 times greater than that in ordinary networks. This enables residual networks to learn complex features. Although the residual networks are deep, their complexity is lower than ordinary networks. Owing to these advantages, residual networks won the 2015 ILSVRC, ImageNet and COCO, all the prestigious competitions in image classification and segmentation.

He et al. made several observations before creating residual networks. First of these observations was on degradation. Degradation problem states that as the number of layers increases, after a certain threshold, the training error increases as well. He et al. observed that most of the time shallower networks perform better than deeper networks. However, in theory, opposite of the observation should occur since a deeper network can be built upon the shallow networks with layers implementing identity maps. He et al. concluded that the learning identity mapping is a difficult task. He et al. also observed that residual mappings are easier to learn in their experiments. Therefore they implemented layers of the networks to learn residual mappings shown in the equation 2.1.

$$H(x) = x + F(x) \tag{2.1}$$

He et al. performed exhaustive tests on residual and other successful models and they discovered that the residual networks have better learning capacity and less computational complexity.

Machine learning has many applications in bioinformatics. Sun et al. created a model on some features they extracted from protein sequence data to predict interactions between proteins [34]. While the study is not about the validation of interfaces, it involves interaction prediction which is similar to the validation. I believe our study

is stronger since we did not use hand driven features and we directly worked on structural data which is more complex than sequence data. In the rest of the subsection, I will explain the features they used, and the model they created.

Sun et al. used two different representations for their data. These representations can be viewed as initial features hand-driven from the protein sequence data. The first one of the representations is autocovariance method, the equation of which is given in 2.2.

$$AC_{lag,j} = \frac{1}{n-lag} \sum_{i=1}^{n-lag} (X_{i,j} - \sum_{i=1}^n X_{i,j}) \times (X_{(i+lag),j} - \sum_{i=1}^n X_{i,j}) \quad (2.2)$$

In equation 2.2, j refers to a specific attribute. In the case of Sun et al., it was 7 for 7 distinct electrochemical properties of the amino acids. “Lag” is the window of interest for autocovariance to be calculated. Sun et al. used a window of 30 amino acids. Thus, lag is 30. i is the position of an amino acid in the sequence and n is the length of the sequence. Since they chose 7 attributes and window size of 30, in the end of the process they had a vector of 7x30 for each protein. They normalized the vectors. They also concatenated vectors of two proteins before giving it to the model for interaction prediction.

The second representation is conjoint triad method. Similar to the autocovariance method, the conjoint triad method is a well studied representation for protein sequences. It involves clustering of amino acids into several groups. In this study, they used 7 clusters based on volumes of dipoles and side chains of the amino acids. After, amino acids are clustered into 7 groups, amino acids in a given protein sequence is changed into the corresponding group numbers. The representation 2.3 shows the example from the paper. Conjoint triad method also requires a window size, in this paper it is 3. They calculated frequency of every possible sequence comprise of 3 amino acids. Calculated 343 (7x7x7) numbers constitute the conjoint triad representation.

$$MREIVHIQAG \longrightarrow 3562142411 \quad (2.3)$$

Sun et al. used a stacked autoencoder to predict interactions. Autoencoder is

a unsupervised algorithm, which does not need any labels. Autoencoders consist of two parts: the first part of the network, called “encoder”, encodes the given input by learning representation of the data by fewer fields. After the input is decoded, the second part of the network, called “decoder”, reconstructs the given input from the representation produced by the encoder part. The loss function of the network compares the given input and the reconstructed input and penalizes any difference. Thus autoencoders learn to represent data with few bits. By stacking more layers, they can learn to represent data with even fewer bits.

Sun et al. used a stacked autoencoder to learn representation from outputs of autocovariance and conjoint triad methods. After the autoencoder generates the representations, a fully connected softmax classifier predicts the possibility of interaction between given proteins. They reported that autocovariance method gives better results than conjoint triad method.

2.3 Scoring Functions

While our model is not a conventional scoring function, it carries out the same tasks in the context of interaction prediction. The conventional scoring functions calculate the energy of the interactions based on the attributes such as electrochemical properties of the amino acids, interactions between solvent and solute and position of the amino acids. We hoped that the model would learn these properties by learning from real interactions.

Chen et al. developed a scoring function for one of the most prominent docking algorithms ZDOCK[4]. In the paper called “ZDOCK: an initial-stage protein-docking algorithm”, they compared four scoring functions to determine the best one to use with their FFT algorithm. These scoring functions were combinations of some minor scoring functions.

One of the first scoring function, commonly used by docking methods, is a grid-based shape complementarity (GSC). Based on GSC, algorithms divide the space where two proteins interact into grids and check for clashes and how well two proteins

filled the gaps without clashing. Clashes are penalized while tightly packed space is rewarded.

For grinds, the space falls into 4 types: solvent accessible surface layer, solvent excluding surface layer, core and open space[5]. Chen et al. determine whether a grid is solvent accessible with “water probing” method. The solvent accessible surface layer of a protein consists of atoms which can directly interact with a water molecule. They also labeled the surface to which no water molecule can access as solvent excluding surface layer. They assigned values to the grids as described in the equation 2.4 [5] for receptor.

$$R_{GSC}(l, m, n) = \begin{cases} 1 & \text{solvent accessible surface layer} \\ 9i & \text{solvent excluding surface layer} \\ 9i & \text{core} \\ 0 & \text{open space} \end{cases} \quad (2.4)$$

l , m , and n are grid indexes in 3 dimensions. i is the imaginary number which is $\sqrt{-1}$. For ligands, values are assigned to grids with few minor changes, shown in the equation 2.5.

$$L_{GSC}(l, m, n) = \begin{cases} 0 & \text{solvent accessible surface layer} \\ 1 & \text{solvent excluding surface layer} \\ 9i & \text{core} \\ 0 & \text{open space} \end{cases} \quad (2.5)$$

After grids are assigned values, the scoring of a docked protein is calculated as described in the equation 2.6.

$$S_{GSC}(o, p, q) = Re[\sum_{l=1}^N \sum_{m=1}^N \sum_{n=1}^N R_{GSC}(l, m, n) \times L_{GSC}(l + o, m + p, n + q)] \\ - Im[\sum_{l=1}^N \sum_{m=1}^N \sum_{n=1}^N R_{GSC}(l, m, n) \times L_{GSC}(l + o, m + p, n + q)] \quad (2.6)$$

While it rewards surface interactions, it penalizes the clashed. While the penalty for surface-core clashes is -9, the penalty for core-core clashes is -81. It rewards every unit of “desired” interaction by 1.

Pairwise shape complementarity (PSC) is an improved version of GSC. PSC's reward system is an improvement on GSC's. PCS rewards the greater number of matched pairs between ligand and receptor proteins. PSC pairs the ligand atoms within a distance of a user determined value plus radius of the receptor atom. As the number of pairs increases, the reward increases. The equations 2.7, 2.8, 2.9 give the grid values for PSC. The score function is given in the equation 2.11.

$$Re[R_{PSC}(l, m, n)] = \begin{cases} \text{number of receptor atoms} & \text{open space} \\ \text{within (D+receptor atom} \\ \text{radius)} & \\ 0 & \text{otherwise} \end{cases} \quad (2.7)$$

$$Re[L_{PSC}(l, m, n)] = \begin{cases} 1 & \text{if this grid is the nearest grid of a} \\ & \text{ligand atom} \\ 0 & \text{otherwise} \end{cases} \quad (2.8)$$

$$Im[R_{GSC}(l, m, n)] = Im[L_{GSC}(l, m, n)] = \quad (2.9)$$

$$\begin{cases} 3 & \text{solvent excluding surface layer} \\ 9 & \text{core} \\ 0 & \text{open space} \end{cases} \quad (2.10)$$

$$S_{PSC}(o, p, q) = Re\left[\sum_{l=1}^N \sum_{m=1}^N \sum_{n=1}^N R_{PSC}(l, m, n) \times L_{PSC}(l + o, m + p, n + q)\right] \quad (2.11)$$

By rewarding pairwise interactions, pairwise shaped complementary scoring works better than grid-based shaped complementary. It is not surprising since proteins naturally interact to pair as much atom as possible to lower their energy.

Desolvation and electrostatics are other two parts of the scoring functions [4]. Chen et al. calculated desolvation using contact energy. Desolvation is calculated as energy required to break protein-water interaction and to form protein-protein interaction. They drove scores for desolvation from a set of previously recorded interaction energies. Combining desolvation with pairwise shape complementary scoring, they reported that the resulting scoring function is a drastic improvement over PSC [4]. To combine two scorings into one, they had to change pairwise shape complementary scoring, since PSC's rewards are positive values and penalties are negative values, whereas desolvation represents rewards as negative values and penalties as positive values. The equations 2.12, 2.13, 2.14 and 2.15 give the grid values and the scoring function.

$$R_{PSC} = L_{PSC} = \begin{cases} 3 & \text{solvent excluding surface layer} \\ 3^2 & \text{core} \\ 0 & \text{open space} \end{cases} \quad (2.12)$$

$$Re[R_{DE}] = Re[L_{DE}] = \begin{cases} \text{sum of PSC and ACE scores of all} & \text{open space} \\ \text{nearby atoms} & \\ 0 & \text{otherwise} \end{cases} \quad (2.13)$$

$$Im[R_{DE}] = Im[L_{DE}] = \begin{cases} 1 & \text{if this grid points is the nearest grid point} \\ & \text{of an atom} \\ 0 & \text{otherwise} \end{cases} \quad (2.14)$$

$$S_{PSC+DE} = Re[R_{PSC} \cdot L_{PSC}] + \frac{1}{2} \times Im[R_{DE} \cdot L_{DE}] \quad (2.15)$$

As the last improvement on scoring function Chen et al. added electrostatic energy to PSC with desolvation. Chen et al. added electrostatic energy using Coulomb's Law. They added the electric potential energy produced by the partial charges of receptor and ligand atoms. The equations 2.16, 2.17, 2.18, and 2.19 give the grid values and the scoring function. Chen et al. reported that the best accuracy is obtained by the combination of pairwise shape complementary, desolvation, and electrostatics.

$$Re[R_{PSC+ELEC}] = Re[L_{PSC+ELEC}] = \begin{cases} 3.5 & \text{solvent excluding surface layer} \\ 3.5^2 & \text{core} \\ 0 & \text{open space} \end{cases} \quad (2.16)$$

$$Im[R_{PSC+ELEC}] = \begin{cases} \beta \times \text{electric potential of open space} \\ \text{all receptor atoms} \\ 0 & \text{otherwise} \end{cases} \quad (2.17)$$

$$Im[L_{PSC+ELEC}] = \begin{cases} -1 \times \text{atom charge} & \text{if this grid points is the nearest grid point of a ligand atom} \\ 0 & \text{otherwise} \end{cases} \quad (2.18)$$

$$S_{PSC+DE+ELEC} = Re[R_{PSC+ELEC} \cdot L_{PSC+ELEC}] + \frac{1}{2} \times Im[R_{DE} \cdot L_{DE}] \quad (2.19)$$

2.4 Data

We sampled our dataset from Protein Data Bank[3] and various datasets created for the development of scoring functions. These datasets are DOCKGROUND[24] and PPI4DOCK[38] for negative interfaces, whereas PIFACE[7] and the structures which we sampled from PDB for positive interfaces.

We selected negative interfaces from PPI4DOCK benchmark. PPI4DOCK has 1417 unbound models which can produce up to 54000 structures each. The selection of 1417 unbound models is a 5-step procedure. In the first step, Yu and Guerois selected 5641 dimeric interfaces, each between two monomeric proteins with less than 70% sequence similarity between from PDB. They also selected 3157 interfaces among initial 5641 interfaces according to some criteria. Resolutions of the selected structures

are less than 3.25 Å. The structures do not contain more than 20 chains. Yu and Guerois classified the interfaces between chains as "biological" and "crystal". The tool which classifies the interfaces as biological/crystal is NOXclass[40]. The area of the interfaces is greater than 300 Å². The interfaces satisfying all of these criteria pass to the next step.

In the second step, Yu and Guerois used HHsearch [32] to find homologs for the both sides of the 3157 interfaces. To be added as templates, found homologs need to comply with certain criteria. The first criteria is that HHsearch probability should be more than 50%. The second criteria is that the sequence similarity between the query chain and the found homolog should be more than 20%. The final criteria which all candidate templates should meet is that the numbers of the aligned columns should be more than 20. After these steps, at most 60 templates pass to the third step.

In the third step, Yu and Guerois selected the best templates among the templates coming from the previous step after another 7 steps. The first step is to eliminate structures with resolution more than 4 Å. After they eliminated the structures with poor quality, they also eliminated the structure with more than 20 biological units. To be able to successfully model interactions, they selected the templates which cover at least 40% of the interface. They compared the structures of the query protein that they selected from PDB and the corresponding templates with TM-align[39]. TM-align gives a score between 0 and 1, a score of 0 means given structures do not bear any similarities and a score of 1 means given structures are the same. Yu and Guerois selected the templates with the score of 0.6 (compared with the corresponding query protein) to make sure they belong to the same family. As a final step, they also checked interfaces with iAlign so that the interfaces are distinct in the templates.

In the fourth step, if the both sides of a heterodimeric interface had a successful template structure, then the interface could be modeled using homology modeling. Yu and Guerois used RosettaCM which includes 3 major steps. It models loop regions and supplants residues to acquire more meaningful structures. It also suggests completion for the missing N-C terminals to the user. After this step, they obtained 1531 modeled

interactions between homologs.

In the last step, Yu and Guerois evaluated the homology models. First of all, they determined few structures with many subunits. They eliminated these subunits to emphasize the interface region more. When Yu and Guerois observed that the stripping of the subunits are changing the structure of the interface, they omitted those models from the dataset. After the initial elimination, the models are aligned using TM-align with the corresponding query proteins. If the query protein and the model have less than 50% sequence similarity, for the template to be included in the dataset, the TM-score should be more than 0.5. However, the sequence similarity of more than 50% signals a good quality modeling without structural evaluation, therefore when the sequence similarity is more than 50%, the model is accepted regardless of the TM-align score.

After creating the homology models using the above mentioned procedure, Yu and Guerois created CAPRI evaluations and the dock structures with ZDOCK[4]. ZDOCK is a FFT (fast Fourier transform) based docking algorithm. Taking advantage of FFT algorithm, ZDOCK exhaustively tries all possible translations and rotations of ligand unit on receptor unit with an unit step in a short time period. The CAPRI evaluations of the docked structures are also available.

DOCKGROUND is an expanding dataset with several subsets for different tasks. These subsets are bound structures, unbound structures, and simulated unbound structures. The decoy sets include structures from unbound structures and simulated unbound structures. The simulated unbound structures are acquired by 1 nanosecond simulations of the extracted chains. Set 1 consists of 61 complexes while Set 2 consists of 351 complexes. Douguet et al. used FFT based tool GRAMM-X to create structures. Each complexes in both sets have 100 structures at least one of which is near native. DOCKGROUND benchmark provides lists of LRMSD, IRMSD, and Fnat values for each structures. They did not include CAPRI evaluations. However, given quantities (LRMSD, IRMSD, and Fnat) can be used to derive CAPRI classifications.

PIFACE[7] data set is our main source of positive interfaces. It consists of in-

terfaces belonging structures published in PDB before 2012. The main aim of the study is to cluster interfaces into unique sets to investigate different types of interface structures and sequences which discovered so far. Cukuroglu et al. used structural alignment tools and network algorithms to achieve clustering.

First of all, Cukuroglu et al. scanned all the PDB files published until January 14th 2012 and found that 45491 of 78477 of the files belong to structure with more than one chains. They reported that if the all binary interactions occurred between all chains in a file, there should be 622321 interfaces. It is a hyperbole, since not all the pairs in complex structure physically interact with each other. Cukuroglu et al. used NACCESS[15] to determine the correct number of the interactions in PDB. NACCESS calculates the accessible surface area of the protein structures. Cukuroglu et al. suggested that if the accessible surface area of a chain as a monomer differs from the accessible surface area of the chain in a complex with a significant amount, the chain should interact with other chains in the complex. Having applied this idea, they discovered 130209 interfaces.

To structurally align 130209 interfaces with each other, Cukuroglu et al. used a structural alignment tool named MultiProt[31]. MultiProt is a tool to align multiple structures. However, in this work, they used it to align pair of structures. Taking the advantage of the geometric hashing algorithm, they perform sequence independent alignment which means sequence information does not affect the alignment. The sequence independent alignment is important in this case because the interfaces do not usually comprise of consecutive residues. Without any sequence alignment, Cukuroglu et al. aligned two interfaces using MultiProt.

To evaluate how similar the given two interface are, Cukuroglu et al. proposed a value of interface similarity. The interface similarity is calculated as:

$$\frac{M}{\min(N_1, N_2)} \quad (2.20)$$

In the definition 2.20, M is the number of residues matched by MultiProt. N variables are the number of the corresponding interfaces. Cukuroglu et al. measured the similarity between interfaces as the ratio between matched residues and the number

of residues of the shorter interface. To avoid too much computation, they eliminated unbalanced pairs of interfaces in which one pair is much longer than the other one.

Cukuroglu et al. used graph algorithms to cluster interfaces. To understand the work done few terms should be explained. Cukuroglu et al. modeled interfaces as nodes. The edges between these nodes represent similarity between interfaces as defined in the equation 2.20. To cluster interfaces, they used Girvan and Newman method [11], in which two definitions are emphasized. The first definition is communities. A community is a group of densely connected nodes. In this work, a community represents a group of structurally similar interfaces. The second definition is in-betweenness. If the in-betweenness value of an edge is high, it means that the number of shortest paths including the edge is great.

Cukuroglu et al. created a huge network with every interfaces with similarity values calculated using MultiProt. By running a recursive algorithm, they eliminated edges with highest in-betweenness values. The recursion continued until a certain terminal condition is satisfied which will be explained. To speed up the algorithm, the interfaces which have very high similarity scores are grouped together. The similarity values of the new nodes, are the maximum of the similarity values of the merge interfaces. Disconnected subgraphs in the end represents the communities, in other words, the clusters of structurally similar interfaces.

The recursion stops when either of the two conditions, which are about to be explained, is satisfied. The first condition is about the size of the communities. If the number of nodes in a community reaches 5, then the recursion stops for that community to avoid getting clusters with less than 5 nodes. The second condition is that the recursion stops if the clustering coefficient is greater than a certain threshold which is set by Cukuroglu et al. The clustering coefficient is a value defined by Cukuroglu et al. as in the equation 2.21.

$$\frac{3 * \text{Number of Triangles on the Graph}}{\text{Number of Connected Triples of Vertices}} \quad (2.21)$$

As a last step Cukuroglu et al. evaluated their clusters using Silhouette index [29].

The equation 2.22 summarizes the index.

$$\frac{b_i - a_i}{\max(a_i, b_i)} \quad (2.22)$$

a_i is the average dissimilarity value of the i^{th} interface with the other interfaces in the same cluster. b_i is the average dissimilarity value of the i_{th} interface with the other interfaces in the other clusters.

Chapter 3

METHODS

In this chapter, we describe two different methods. The first method, QuickRet, is a filtering tool which ranks interface structures for a given protein structure based on structural similarity with the help of structural alignment. The structural alignment is the process of finding similar 3 dimensional patterns in the structures of the given proteins and minimizing the distance between these patterns. After we describe QuickRet, we will also describe our deep learning algorithm which predicts the probability of an interaction from the given structure.

3.1 *QuickRet*

There are two phases in the algorithm QuickRet. In the first phase, the algorithm creates a hash table from the features extracted from the interfaces in the database. In the second phase, the algorithm searches the database of interfaces with the help of the hash table, created in the first phase, for patterns similar to the ones the query protein has. When the algorithm finds an interface which has features similar to the query protein, it calculates the translation and rotation parameters to align the interface and the query protein. In the end, the algorithm outputs a list of similar interfaces with the transformation parameters to align them. The algorithm sorts the output list according to the number of similar features. Creating hash table is summarized in the algorithm 1. The retrieval part with its two phase is also summarized in the algorithm 2.

3.1.1 Feature Set and Base Set

The feature set is an important part of QuickRet. The feature type directly affects the memory complexity and the accuracy of the algorithm. We used the features used in SnapDock [10]. We calculated the features we used in the hash table from four points coming from two amino acids which have C_α and C_β atoms. We discarded glycines. Alternatively, a C_β atom with a constant angle and a constant distance can be added for every glycine. The algorithm requires the two C_α atoms to be no closer than 4 Å and no further than 13 Å. We define any four points which satisfy the distance constraint as a base. Thus, a base is a set of 12 numbers, xyz coordinates of the chosen 2 C_α atoms and 2 C_β atoms. From every base, the algorithm calculates two bond angles, one torsion angle and the distance between two C_α atoms. The features and the base are shown in the Figure 3.1. In the Figure 3.1, the angles ‘a’ and ‘b’ are the bond angles, the ‘w’ is the torsion angle while the ‘d’ is the distance between two C_α . The equations 3.1, 3.2 show how to calculate bond angles ‘a’, ‘b’ and torsion angles ‘w’ consecutively:

$$\begin{aligned}
 \overrightarrow{C_{\alpha 1}C_{\beta 1}} &= C_{\beta 1} - C_{\alpha 1} \\
 \overrightarrow{C_{\alpha 1}C_{\alpha 2}} &= C_{\alpha 2} - C_{\alpha 1} \\
 \angle a &= \arccos(\langle \overrightarrow{C_{\alpha 1}C_{\beta 1}}, \overrightarrow{C_{\alpha 1}C_{\alpha 2}} \rangle)
 \end{aligned} \tag{3.1}$$

$$\begin{aligned}
 \overrightarrow{C_{\beta 1}C_{\alpha 1}} &= C_{\alpha 1} - C_{\beta 1} \\
 \overrightarrow{C_{\alpha 1}C_{\alpha 2}} &= C_{\alpha 2} - C_{\alpha 1} \\
 \overrightarrow{C_{\alpha 2}C_{\beta 2}} &= C_{\beta 2} - C_{\alpha 2} \\
 \vec{u} &= \frac{\overrightarrow{C_{\beta 1}C_{\alpha 1}} \times \overrightarrow{C_{\alpha 1}C_{\alpha 2}}}{\|\overrightarrow{C_{\beta 1}C_{\alpha 1}} \times \overrightarrow{C_{\alpha 1}C_{\alpha 2}}\|} \\
 \vec{v} &= \frac{\overrightarrow{C_{\alpha 1}C_{\alpha 2}} \times \overrightarrow{C_{\alpha 2}C_{\beta 2}}}{\|\overrightarrow{C_{\alpha 1}C_{\alpha 2}} \times \overrightarrow{C_{\alpha 2}C_{\beta 2}}\|} \\
 \angle w &= \text{sign}(\langle \vec{u}, \overrightarrow{C_{\alpha 2}C_{\beta 2}} \rangle) \cdot \arccos(\langle \vec{u}, \vec{v} \rangle)
 \end{aligned} \tag{3.2}$$

$\vec{u}$ and $\vec{v}$ are the unit normals of the planes determined by the points $C_{\beta 1}$, $C_{\alpha 1}$, $C_{\alpha 2}$ and the points $C_{\alpha 1}$, $C_{\alpha 2}$, $C_{\beta 2}$ consecutively. The base in the Figure 3.1 is the list of the

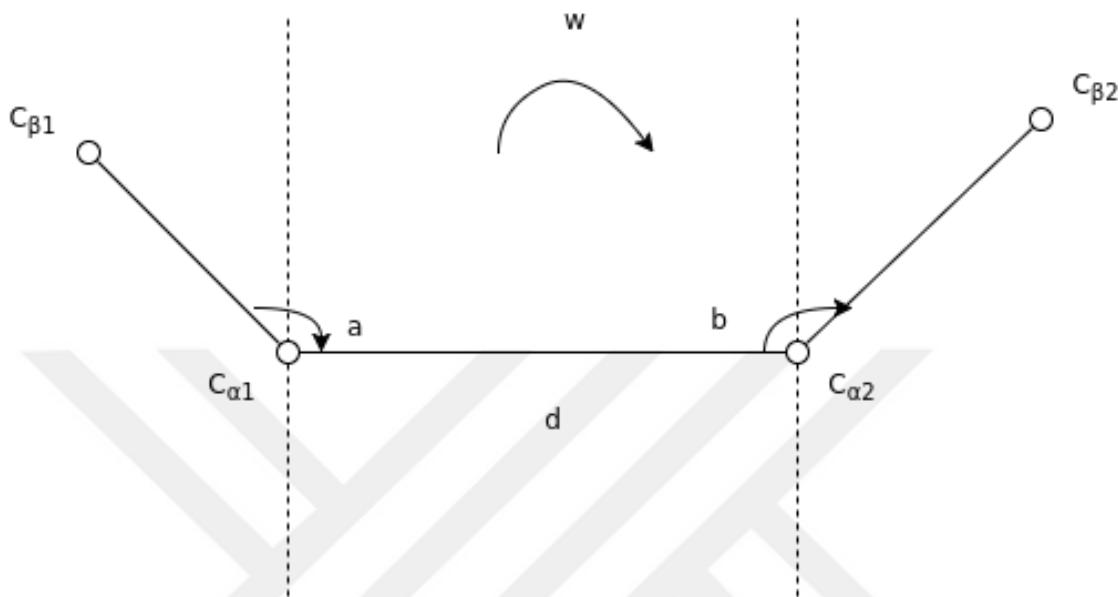


Figure 3.1: The features and the base.

coordinates of the atoms $C_{\beta1}$, $C_{\alpha1}$, $C_{\alpha2}$, $C_{\beta2}$; [Coordinate of $C_{\beta1}$, Coordinate of $C_{\alpha1}$, Coordinate of $C_{\alpha2}$, Coordinate of $C_{\beta2}$].

3.1.2 Creating Hash Table

The algorithm 1 summarizes the hash table creation process of the QuickRet. The hash table construction requires a list of interfaces and threshold parameters. When the user provides the interface database and the thresholds which will be used by the algorithm to discretize the bond angles, the torsion angle and the distance between two C_{α} , the algorithm starts iteration over all interface structures.

First, we need to describe discretization. After the user provides the thresholds for the distance, the bond angles and the torsion angle, the algorithm uses them to create the hash table. We will show the discretization process using only one element of the feature set, the torsion angle, since the process is identical for the rest of them. After the torsion angle is calculated as described above, the torsion angle is divided by the corresponding threshold and then the result is rounded to the closest smaller integer value. For example, if the torsion angle is 130° , and the threshold for the torsion angle is 20° , then the discretized value for the torsion angle is 6 because the closest smaller

integer value to $130/20 = 6.5$ is 6. After all features are discretized, the new set of 4 integers are defined as a key for the hash table.

After an interface structure is read, the algorithm assigns a model number to the structure. For each interface, the algorithm calculates the features. The algorithm discretizes the values in the feature set to use them as a key for the hash table as described above. The algorithm inserts the base, b in the pseudo code 1, and an identification number, i in the pseudo code 1, which specifically belongs to a model. Thus we can visualize a bin as $hashtable(key) = [(b_1, i_1), (b_2, i_2), (b_3, i_3) \dots]$. The algorithm completes creating the hash table when it extracts the features from every interface in the database and store them in the hash table. In the end, the hash table has lists of pairs of bases and model identification numbers in each bins which can be accessed by a discretized feature set.

Data: List of interface structures

Result: Hash table of features extracted from the interface structures

initialize hashtable;

for *interface in the interface list* **do**

 read interface from PDB file;

 assign a model number i to interface;

 list of feature-base pairs $\leftarrow$ **extractFeatures**(interface);

for f, b in list of features-base pairs **do**

 key $\leftarrow$ discretize f ;

 insert (b, i) into hashtable(key);

end

end

Algorithm 1: Creating the hash table.

3.1.3 Retrieval

The pseudo code of the retrieval process is given in the algorithm 2. For the second phase, the algorithm requires a query protein and a hash table created in the first phase. QuickRet extracts the same features for the query protein as described above.

After the extraction, the algorithm discretizes the feature set using the threshold parameters to obtain a key for the hash table. Each key has a possibility to retrieve a list of base-model pairs from the hash table; $hashtable(key) = [(b_1, i_1), (b_2, i_2), (b_3, i_3) \dots]$. If the key is a hit, the bin consisting of base-model pairs are retrieved. Algorithm inserts the base from the query protein, b_q in the algorithm 2, which the algorithm uses to calculate the key, paired with the bases retrieved from the hash table, $b_{interface}$ in the algorithm 2, into a new hash table hashed by the model identification number of the interfaces. In the algorithm 2, we named the new hash table “matchlist” and we show the model identification number as “i.” At the end of the process, the algorithm produces a dictionary of models and their bases matched with the bases of the query protein. Thus a bin consists of the matched pairs of bases coming from the query protein and a specific interface in the database. Let say the interface’s model identification number is i . We also define the bases coming from the query protein which has n bases as $base_{qk}$, k ranges from 1 to n . Finally, we define the bases coming from the interface which as m bases as $base_{intl}$, l ranges from 1 to m . Now, we can give an example for a bin of “matchlist”:

$$matchlist(i) = [(b_{q1}, b_{int1}), (b_{q1}, b_{int3}) \dots, (b_{q2}, b_{int9}), (b_{q2}, b_{int10}) \dots (b_{qn}, b_{int1}) \dots].$$

The algorithm votes every model in the retrieved list. Every match in the list is a vote for the model. The algorithm later uses these votes to calculate a score for the interface structure. The algorithm proceeds by iterating over the dictionary. For every matching base, the algorithm calculates the transformation parameters to superpose the bases extracted from the interface structure onto their matching bases extracted from the protein structure. For example, from the example above, the algorithm calculates the translation and rotation parameters to superpose b_{int1} onto b_{q1} . How to calculate these parameters are described in the section “Transformation Parameters.” After transformation parameters are extracted, we have 6 transformation parameters, two for each axis. The algorithm obtains a key for the second hash table by discretizing 6 transformation parameters. This time above discretization process is applied without thresholds. The parameters are rounded to the closest smaller in-

teger values. We inserted transformation parameters into a 6 dimensional hash table by using the keys.

Data: Query Protein

Result: List of Poses

read protein structure from PDB file;

list of feature-base pairs $\leftarrow \text{extractFeatures}(\text{protein})$;

initialize matchlist;

for f, b_q *in* list of feature-base pairs **do**

 key $\leftarrow$ discretize f ;

 retrieve $(b_{\text{interface}}, i)$ pairlist from hashtable(key);

for $(b_{\text{interface}}, i)$ *in* pairlist **do**

 insert $(b_{\text{interface}}, b_q)$ into matchlist[i];

end

end

initialize bestposes;

initialize posetable;

score₁ $\leftarrow \text{calcScore}(\text{matchlist})$;

for i *in* matchlist **do**

for $(b_{\text{interface}}, b_q)$ *in* matchlist[i] **do**

 transformation $\leftarrow \text{calcTransformation}(b_{\text{interface}}, b_q)$;

 key $\leftarrow$ discretize transformation;

 insert transformation into posetable(key);

end

 retrieve the pose with most vote and score₂ from posetable;

 insert the pose into bestposes;

end

sort bestposes with score₁ + score₂;

Algorithm 2: Retrieval part

Voting procedure to determine the best transformation between the interface and the protein is very similar to the voting procedure described for the first part. The

algorithm selects the bin with the most transformation. We calculated the average of the each parameters in the bin to output the final transformation. This process is done for every interface whose bases are matched with the ones of the query protein.

3.1.4 Scoring

We designed a heuristic approach to rank the transformations. We combined two scores, first one coming from the first voting, second one coming from the second voting. After the first voting, the algorithm calculates the ratio of how many votes the model got to the total number of features the model has. For example, for the example above, the first part of the score for the interface with the model identification number ‘i’ is $\frac{\text{lenght of } matchlist(i)}{\text{total number of features coming from model i}}$. In the pose clustering phase, the algorithm calculates the ratio of the number of the transformations in a bin to the total number of transformations. The two scores are combined by addition. The algorithm sorts the results according to the score in descending order.

We also used an alternative scoring in the result section. The alternative scoring is given as $Score = (1.1 - \frac{MatchedResidues}{InterfaceResidues}) \times RMSD$. As RMSD of an alignment decreases and the percentage match increases, the score of the interface decreases. In this case, a lower score is preferred for an interface structure.

3.1.5 Transformation Parameters

$$C = A^T B \tag{3.3}$$

$$C = U S V^T \tag{3.4}$$

$$t = \det(UV^T) \tag{3.5}$$

$$R = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & t \end{bmatrix} U^T \tag{3.6}$$

The algorithm keeps the bases as 4 by 3 matrices in the memory. Every row of the matrix is a coordinate of an atom. The point sets are superposed using Kabsch

algorithm [17]. The Kabsch algorithm minimizes the square root deviation between two point sets by using singular value decomposition. The equations 3.3, 3.4, 3.5, and 3.6 describes the steps of Kabsch algorithm. In the equations, the matrix A represents the base coming from the interface and the matrix B represents the base coming from the query protein. The step shown by the equation 3.5 is carried out to check if the coordinate system is right handed and if not to convert it to right handed coordinate system. Kabsch algorithm outputs a rotation matrix. For Kabsch algorithm to work,

the center of the point clouds should be translated to the origin.

$$R(\theta_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & \sin(\theta_x) \\ 0 & -\sin(\theta_x) & \cos(\theta_x) \end{bmatrix} \quad (3.7)$$

$$R(\theta_y) = \begin{bmatrix} \cos(\theta_y) & 0 & -\sin(\theta_y) \\ 0 & 1 & 0 \\ \sin(\theta_y) & 0 & \cos(\theta_y) \end{bmatrix} \quad (3.8)$$

$$R(\theta_z) = \begin{bmatrix} \cos(\theta_z) & \sin(\theta_z) & 0 \\ -\sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (3.9)$$

$$R(\theta_x)R(\theta_y)R(\theta_z) = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} \quad (3.10)$$

$$m_{11} = \cos(\theta_y) \cos(\theta_z) \quad (3.11)$$

$$m_{12} = \cos(\theta_y) \sin(\theta_z) \quad (3.12)$$

$$m_{13} = -\sin(\theta_y) \quad (3.13)$$

$$m_{21} = \sin(\theta_x) \sin(\theta_y) \cos(\theta_z) - \cos(\theta_x) \sin(\theta_z) \quad (3.14)$$

$$m_{22} = \sin(\theta_x) \sin(\theta_y) \sin(\theta_z) + \cos(\theta_x) \cos(\theta_z) \quad (3.15)$$

$$m_{23} = \sin(\theta_x) \cos(\theta_y) \quad (3.16)$$

$$m_{31} = \cos(\theta_x) \sin(\theta_y) \cos(\theta_z) + \sin(\theta_x) \sin(\theta_z) \quad (3.17)$$

$$(3.18)$$

$$m_{32} = \cos(\theta_x) \sin(\theta_y) \sin(\theta_z) - \sin(\theta_x) \cos(\theta_z) \quad (3.19)$$

$$m_{33} = \cos(\theta_x) \cos(\theta_y) \quad (3.20)$$

We calculated translation parameters by subtracting the coordinate of center of the base extracted from the interface from the the coordinate of the center of the base extracted from the query protein. We defined these parameter as t_x , t_y , t_z . We

used the Euler angle extraction method to calculate rotation angles along the x, y and z axes [8]. The equations 3.7, 3.8 and 3.9 show the rotation matrix along the x-axis, the rotation matrix along the y-axis and the rotation matrix along the z-axis consecutively. When we combine rotation matrices, we obtain the rotation matrix presented in the equation 3.10. We used the equations shown in 3.11 to 3.20 to extract Euler angles from the rotation matrix produced by Kabsch algorithm.

Algorithm extracts the rotation angle along the axes by using *atan2*. We used *atan2* function in our calculation instead of *arctan* function because the first one is able to identify the quadrants while the latter cannot. The algorithm first extracts the rotation angle along the x-axis as shown in equations 3.21 and 3.22.

$$\theta_x = \arctan 2(m_{23}, m_{33}) \quad (3.21)$$

$$= \arctan 2(\sin(\theta_x) \cos(\theta_y), \cos(\theta_x) \cos(\theta_y)) \quad (3.22)$$

To extract the rotation angle along the y-axis, the algorithm first calculates the $\cos(\theta_y)$ from the matrix 3.10. The steps of cosine calculation are shown in equations 3.23 to 3.26.

$$\cos(\theta_y) = \sqrt{m_{11}^2 + m_{12}^2} \quad (3.23)$$

$$= \sqrt{\cos(\theta_y)^2 \cos(\theta_z)^2 + \cos(\theta_y)^2 \sin(\theta_z)^2} \quad (3.24)$$

$$= \sqrt{\cos(\theta_y)^2 (\cos(\theta_z)^2 + \sin(\theta_z)^2)} \quad (3.25)$$

$$= \cos(\theta_y) \quad (3.26)$$

Using $\cos(\theta_y)$, the algorithm calculates the rotation angle along the y-axis (θ_y) as shown in the equation.

$$\theta_y = \arctan 2(-m_{13}, \cos(\theta_y)) \quad (3.27)$$

$$= \arctan 2(\sin(\theta_y), \cos(\theta_y)) \quad (3.28)$$

Finally θ_z is calculated as

$$\theta_z = \arctan 2(\sin(\theta_x)m_{31} - \cos(\theta_x)m_{21}, \cos(\theta_x)m_{22} - \sin(\theta_x)m_{32}) \quad (3.29)$$

In the end, the transformation set for a pair of an interface base and a protein base is $\{t_x, t_y, t_z, \theta_x, \theta_y, \theta_z\}$

3.1.6 Data

We used PIFACE [7] to construct our hash table. The PIFACE dataset has 130209 interfaces. Cukuroglu et. al classified the interfaces in the dataset into 22604 clusters. We used the representatives of the 22604 clusters to construct the hash table. The algorithm considers each side of an interface separately, thus there are 45208 candidate structure to construct our hash table. However, we did not use every structure, instead, we eliminated the structures which are smaller than 10 residues. Another problem with the dataset was the obsolete structures. Some structures were omitted from the PDB since 2012, when the dataset was created. After the omission of the improper structures, the algorithm constructs the hash table with 43453 interface structures.

3.1.7 Complexity

Here we present a brief complexity analysis of QuickRet. We assume that the length of the protein is m and the number of the interfaces in a database is n .

For the first part of QuickRet, the algorithm iterates over pairs of residues coming from every interface structures in a database. The length of the interface structures varies between 1 - 574 with a mean of 44.5 and a standard deviation of 31.4. Thus we can assume the length of the interfaces constant. Since there is a distance constraint and we can assume the lengths of interfaces constant, we can also assume that the number of the pairs which satisfy this constraint is constant. For the first part, we have n interfaces in a database. Pairs from these interfaces are inserted in a list in a hash table. Finding a bin in the hash table takes constant time on average and inserting an element to a list takes amortized constant time because of the reallocation. Even if the whole features points out to the same bin, or the user chooses a threshold big enough so that every feature goes into the same bin, the complexity of the first part turns out to be $\mathcal{O}(n)$.

For the second part of QuickRet, we are assuming that the length of the protein sequence is m . For each pair of the matching bases, the algorithm performs operations

in constant time to calculate the transformation parameters. From the first part of the algorithm, we have $O(n)$ bases in a bin. We can again assume that the number of the pairs coming from the query protein is constant. Thus we can give the complexity of the filtering part of the QuickRet as $\mathcal{O}(mn)$.

3.2 Interface Prediction with Deep Learning Methods

3.3 Models

We tried 2 different models to predict interactions from docked structures. One of the model we tried is a simple CNN model, shown in the Figure 3.2. The model is very similar to LeNet[22]. The model has 4 layers. The first two layers are convolutional units which consist of $5 \times 5 \times 5$ kernels. The first one of the convolutional layers includes 50 kernels while the second one includes 60 kernels. After each convolutional layer, we applied the rectifier unit to the output. After rectifier units, we also applied dropout layer to avoid overfitting the data. We set the dropout probability of an unit to 0.7 due to the tendency of the network towards overfitting. Finally, after the dropout, we applied pooling to the output. The pooling is maximum with no padding and the window size is 2.

The last two layers are fully connected units. The first fully connected unit consists of 920 unit while the second fully connected unit consists of 2 units. We applied the rectifier units to the outputs of the both fully connected layers. After the rectified linear units, we applied dropout with the probability of 0.7, similar to the first two layers. After the second fully connected layer, we applied softmax function to the probability distribution over two possibilities. After softmax function, the first value shows the possibility of an interaction.

The second model is a deeper model than the previous model, shown in the Figure 3.3. We doubled the convolutional layers compared to the first model. The 4 convolutional layers have $5 \times 5 \times 5$ kernels. There are 128 kernels in each layers. Thus the number of channels is preserved during the execution. We set the stride and the

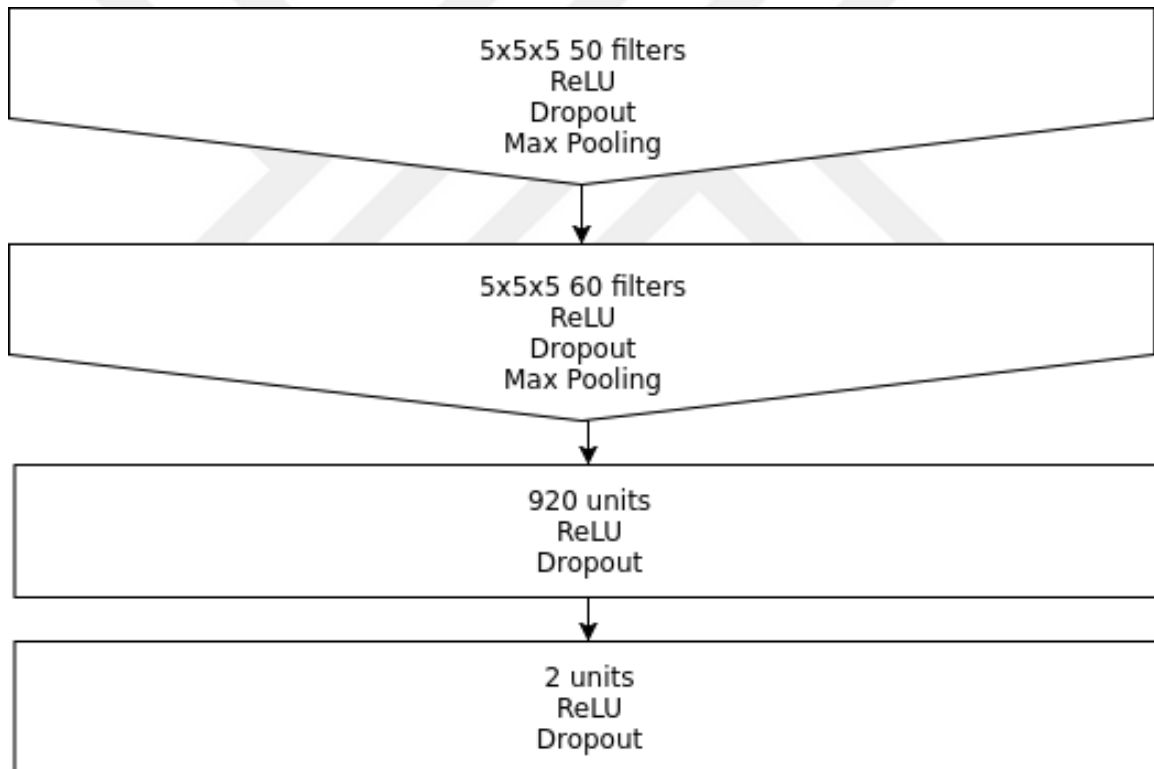


Figure 3.2: Architecture of the first model.

padding size to 2 for every convolutional layer. By making convolutional part of our model deeper, we can get better results since our data is sparse. We applied batch normalization and rectified linear units after the each convolutional layers. We also applied global average pooling after the convolutional layers to avoid overfitting and to generalize better. After the convolutional part, we applied two layers of fully connected units, similar to the first model. First fully connected layer have 512 units. After the first fully connected layer, we applied batch normalization and rectified linear units similar to the convolutional part. The last layer has 2 units. After the softmax function, the two values, the model gives, are the complementary possibilities of the given structures being an interface and not an interface.

3.4 Weight Initialization and Training

We trained our models with Adam [18] optimizer with learning rate of 0.001. We separated our data into minibatches with 32 samples. At the beginning of the each epoch, the training data is shuffled. As the validation loss increases instead of decreasing between consecutive epochs, the learning rate is decayed with the factor of 2.

The weights are initialized using Xavier[12] and Kaiming[13] distributions. We initialized the weights of the convolutional layers with Kaiming normal distribution and the weights of the fully connected layers with Xavier normal distribution.

3.5 Data

We collected data for training and testing our model from 4 sources: DOCKGROUND [24], PPI4DOCK [38], PIFACE [7], and Protein Data Bank (PDB) [3]. The most of the positive interfaces are taken from PIFACE which is a collection of interface structures extracted from PDB files published until 2012. We obtained rest of our positive dataset from protein structure files published in Protein Data Bank after 2012 to avoid duplicates. The other 2 data sets, DOCKGROUND and PPI4DOCK, provides the negative interfaces.

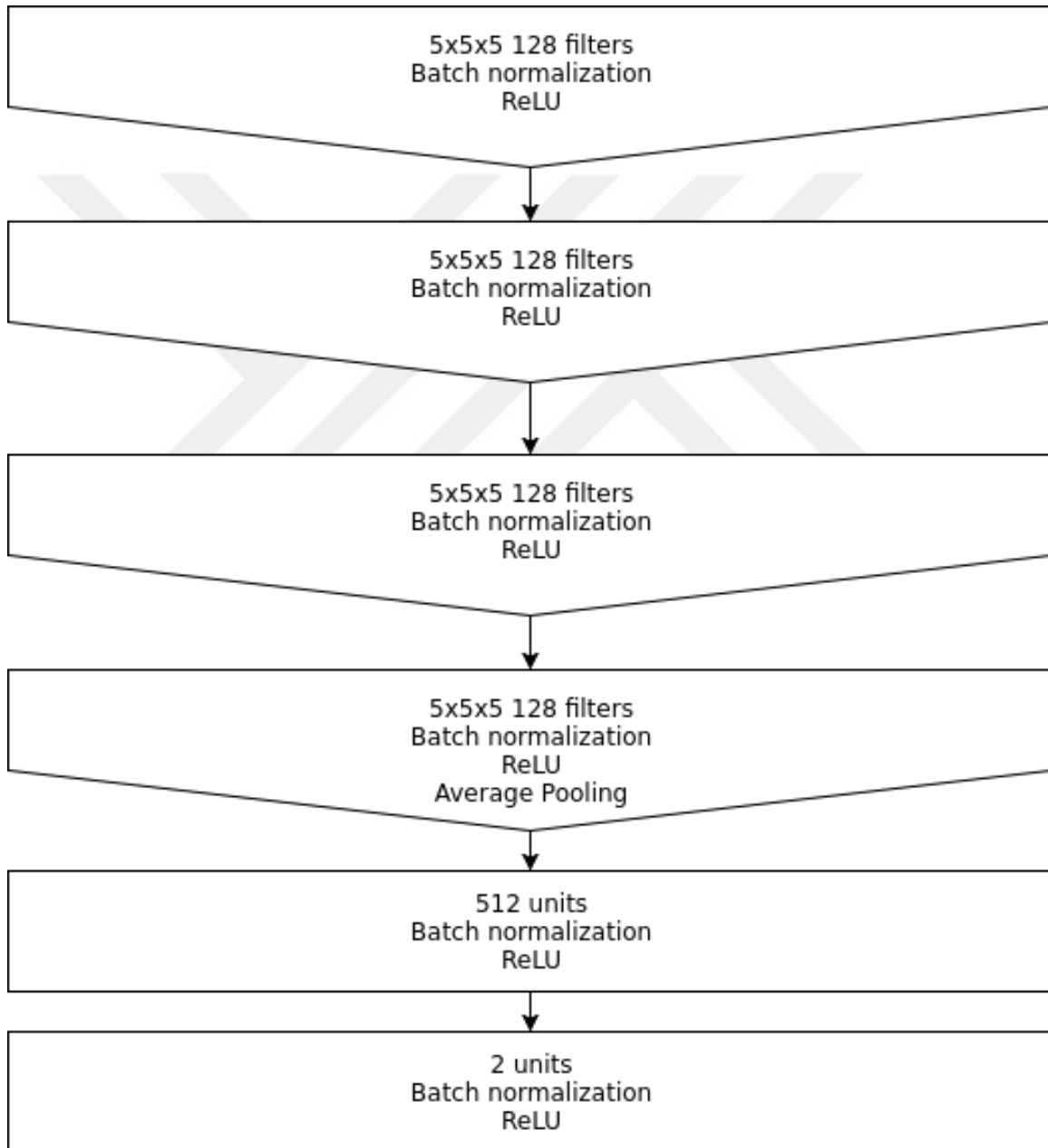


Figure 3.3: Architecture of the second model.

Before explaining the data preparation and data sets, what is meant by “positive” and “negative” interface should be clarified. We consider an interface to be positive if it comes from a biological complex whose structure is documented in Protein Data Bank. To determine whether an interface is negative, we used the classification developed by Critical Assessment of Predicted Interactions (CAPRI) [16]. Within the scope of CAPRI, predicted complexes are evaluated into 4 categories based on 3 quantities: Fnat (fraction of native contacts), LRMSD (ligand root mean square deviation), and IRMSD (interface root mean square deviation). If Fnat of a complex is lower than 0.1 or its IRMSD is greater than 10 Å while its iRMSD is greater than 4 Å, the complex is classified as “incorrect”. We consider interfaces obtained from “incorrect” complexes as negative interfaces. Interface extraction from positive and negative complexes is same. For a complex, residues in one chain which are closer than 12 Å to one or more residue in another chain constitute an interface.

We selected 149557 negative interfaces from the PPI4DOCK benchmark. PPI4DOCK has 1417 unbound models which can produce up to 54000 structures each. PPI4DOCK provides CAPRI assessment of each structure. These models, consisting of 2 chains, are heterodimers. To create structure from models, ZDOCK [4] docking tool is used. ZDOCK being an FFT based algorithm exhaustively tries all possible translations and rotations of ligand unit on receptor unit with a unit step. Taking advantage of the exhaustive search carried out by ZDOCK, we selected approximately 110 incorrect structures with the ligand being in different positions from each model. After the selection, the interfaces are extracted as described above.

DOCKGROUND is an unbound benchmark very similar to PPI4DOCK. It has two separate decoy sets. Set 1 consists of 61 complexes while Set 2 consists of 351 complexes. To create structures, another FFT based tool GRAMM-X [37] is used. Each complex in both sets have 100 structures at least one of which is near native. DOCKGROUND benchmark provides lists of LRMSD, IRMSD, and Fnat values for each structure. We selected structures satisfying CAPRI's incorrect criteria and applied the same interface extraction procedure above.

Table 3.1: Distribution of negative and positive samples.

	PIFACE	PPI4DOCK	DOCKGROUND	PDB2012	Total
Training	127790	127790	-	-	255580
Validation	-	-	4000	4000	8000
Test	-	-	4125	4125	8250
Total	127790	127790	8125	8125	271830

The PIFACE dataset is our main source of positive interfaces. The dataset consists of approximately 130000 interfaces extracted from approximately 80000 PDB files. Cukuroglu et al. clustered these interfaces into approximately 26000 groups using structural similarities. We included already extracted interfaces from PIFACE in our positive dataset. To enrich our positive dataset, we also extracted 16249 interfaces from PDB files which are published after 2012.

After extraction and preparation of the interfaces from all the data sets, we have approximately 300000 interfaces to train and test our model. 156817 interfaces in the database are positive while rest of the data set, 144039, is negative. We separated PIFACE and PPI4DOCK to train our models, and DOCKGROUND and PDB files published after 2012 to test our models. Reason behind separation is to see how well our model generalizes to structures coming from different models. To summarize, our training data set comprises 127790 negative interfaces coming from PPI4DOCK and 127790 positive interfaces coming from PIFACE, our test and validation data sets comprise 8125 positive interfaces from PDB files published after 2012, and 8125 negative interfaces coming from DOCKGROUND. This is summarized in the table 3.1. We also tested our models, using a second dataset which is also shown in the table 3.2.

The procedure which is about to be explained is applied to all data sets to obtain interfaces in a necessary data format to train and test the model. First, the PRISM's [1] template generator extracts interfaces. Since only interfaces are needed, not hotspots or contacts, other modules of the PRISM is not used. After interface

Table 3.2: Alternative distribution of negative and positive samples.

	PIFACE	PPI4DOCK	Total
Training	97790	97790	195580
Validation	15000	15000	30000
Test	15000	15000	30000
Total	127790	127790	255580

extraction, since CNNs are not rotation invariant, we first rotated all interfaces to give them similar poses. We translated the midpoints between the centers of mass of the two side of the interfaces to the origin and rotated the centers of mass onto x axis. We also rotated the interfaces so that the atom which is furthest from the origin, after the first transformation, points positive z direction. After the rotation and translation steps, 40 by 25 by 25 by 25 matrices are created for every interface. The CNN model works on 3 dimensional matrices with 40 channels: one channel each for the side chains of the each chains' every 19 residue types excluding GLY and 1 channel each for the alpha Cs. The shape is determined so that every cell represents $2 \times 2 \times 2 \text{ \AA}^3$ of $50 \times 50 \times 50 \text{ \AA}^3$ of space which includes the interface. If an interface cannot fit into $50 \times 50 \times 50 \text{ \AA}^3$ with its given position, that interface is eliminated from the data set. The amino acids are considered as 3D Gaussian distributions with the coordinates of amino acids as the means. The standard deviation of the distributions are derived from the Van der Waals radii of the each amino acid types. Distributions are quantized to fill cells of matrices.

Chapter 4

RESULTS AND DISCUSSION

In this chapter, we will describe the experiments we performed. After we describe the experiments, we will present our results and analyze these results. In the first section, we will present the results for QuickRet. In the second section, we will present the results for our deep learning method for interface prediction. Before we present our results, we need to make a clarification about our RMSD values. Since we did not find correspondence with our algorithm, the correspondence is found afterwards by matching residues which are closer than 3.8 Å to each other. Thus, our RMSD values cannot pass 3.8 Å. Therefore we also check the percentage of interfaces which are matched with the proteins to better analyze our results. Also we are not comparing our algorithm with any alignment tools based on RMSD values.

4.1 *QuickRet*

We subjected QuickRet to 6 different tests. In the first test, we checked the alignment of the first ranked interface with the query protein. In the second test, we compare QuickRet with TMalign, a pairwise alignment tool by ranking first 20 interface structures QuickRet found with the RMSD scores of the alignments produced by TMalign. In the third test, we used 166 proteins from the PIFACE database. These 166 proteins are selected randomly among the members of the clusters with more than 5 proteins. We checked how QuickRet ranked the representative when we query an interface structure for a member of their cluster. We also ranked the outputs using the percentage of the matched residues and RMSD values to check how well the QuickRet scores the alignments. Finally, in the fourth test, we used the proteins from the prediction made by PRISM. We selected 82 proteins randomly among the predic-

tions of PRISM. We checked how QuickRet ranked the predictions made by PRISM. Again, we ranked the results of QuickRet using RMSD and percentage match values to compare with QuickRet’s scoring. We also evaluate how fast QuickRet compared to using TMalign to rank interface structures.

4.1.1 Test 1: Top Matching Interfaces

The results of the first test is shown in the table 4.1. The name of the interface structures are given in the first column. The first 4 letters represents the PDB code from which the interface is extracted. The 5th and 6th letters are the interacting chains. The letter after the underscore tells which side of the interface is being used. For example, “1zaxAZ_Z” is extracted from “1zax” structure and the structure is the Z chain’s side of the interface between A chain and Z chain. We listed the chains in the second columns. The first 4 letters belong to PDB id while the last letter represents the chain. Thus “1twiD” is the D chain of the structure named “1twi.” The fourth and the sixth columns are the number of residues in the interface structure and the number of residues in the protein structure. The third column is the correctly matched number of residues. We determined whether a match is correct by checking if the distance between two residues is less than 3.8 Å [19]. We listed ratio of the number of the correctly matched residues and the number of interface residues in the fifth column. Finally, the seventh column shows the RMSD values.

Table 4.1: Alignment of top 1 interface structures with the given proteins.

CM: Correct matches IL: Interface length in residues PM: percentage of interface matched with the protein PL: Protein length in residues

Interface	Chain	CM	IL	PM	PL	RMSD
1zaxAZ_Z	1twiD	8	17	0.47	824	2.3827
3cbaCJ_C	3nenB	16	23	0.70	476	2.50283
1zaxAZ_Z	1s0wA	5	17	0.29	333	2.25025
2izxBC_C	3e9nB	9	18	0.5	205	2.71103

3cbaCJ_J	2aunA	11	23	0.48	450	1.96747
1n2dAC_C	1folA	23	30	0.77	597	1.26173
3kurAF_A	8rucA	6	8	0.75	783	2.59108
2vs6AB_B	1ws5G	10	39	0.26	192	2.62869
1s5aBD_B	3a20A	13	36	0.36	260	1.94257
1s17AB_B	1jywC	7	13	0.54	2152	2.12057
3cbaCJ_J	1zpdB	17	23	0.74	1214	1.50251
2ahmBG_G	3i5uA	14	22	0.64	458	2.35136
3robAD_A	2i8aD	7	18	0.39	357	2.07829
2vpnAB_A	3rufS	1	20	0.05	439	3.38187
1zaxAZ_Z	2bmcF	9	17	0.53	295	2.59324
3cbaCJ_C	2jd8B	6	23	0.26	183	2.77193
1zaxAZ_Z	1n2cD	8	17	0.47	524	2.27408
3cbaCJ_J	3qd8G	6	23	0.26	181	2.34878
3d6bBD_D	3kopB	23	41	0.56	299	2.20291
3lhbCG_G	1ykmC	10	14	0.71	278	2.23518
1hzdBf_B	2vdhL	18	28	0.64	196	0.813714
3h43IK_I	3gxfB	9	15	0.6	762	2.77176
1zaxAZ_Z	1efhA	11	17	0.65	375	2.61076
3a1qDF_F	1hfbE	11	21	0.52	437	2.21143
2cioAB_B	1soxB	4	5	0.8	869	2.2574
3favAC_A	1j2tB	5	16	0.31	496	2.08725
3favAC_A	2ba2C	6	16	0.37	144	2.26117
3u1aBC_C	1aonD	2	16	0.12	526	2.96292
3favAC_A	3n5mC	10	16	0.62	738	2.16528
3cbaCJ_C	1kxvA	10	23	0.43	1086	2.59675
2ww4AB_A	2vutI	6	54	0.11	64	2.43224
2ckbDI_I	2ckbH	25	26	0.96	275	1.99608
2f4iAB_B	1qxsC	9	14	0.64	436	2.29321

3favAC_C	3mk7L	13	16	0.81	198	1.76093
1lghAD_A	1kzuH	21	39	0.54	49	1.3597
3qp5AC_C	2gttC	5	15	0.33	404	1.77459
3favAC_C	1ho4A	4	16	0.25	438	2.83581
1hv2AB_B	3tdkL	9	13	0.69	488	2.46218
2dvrBQ_Q	2c5bA	4	9	0.44	360	2.2192
1zaxAZ_Z	1zumC	11	17	0.65	762	2.70153
3kzeCE_E	2exjB	4	6	0.67	1007	2.78658
3cbaCJ_J	3is7B	1	23	0.04	233	2.06918
3favAC_C	3d29A	8	16	0.5	305	2.21908
3p57BP_P	3rycC	8	18	0.44	581	1.86356
1gkdAB_B	1l9pA	9	33	0.27	820	2.33069
3batAC_C	3o7mB	7	25	0.28	279	2.85468
2x89BD_B	3m24B	15	26	0.58	356	2.47544
2f4iAB_B	2f6pA	8	14	0.57	443	2.28513
3rk2BD_B	1xinA	21	29	0.72	614	1.1007
3lnzCI_I	2h53T	13	24	0.54	93	2.05874
3degDE_D	2hwnE	12	45	0.27	32	2.06102
3cbaCJ_C	2y70B	6	23	0.26	402	2.26952
2dvrBQ_Q	1wbxA	1	9	0.11	622	1.60699
1zaxAZ_Z	3nzxV	8	17	0.47	269	2.48145
3subAB_A	2rmu4	4	39	0.10	48	2.51096
3favAC_A	3nnkF	9	16	0.56	483	1.98275
1y4mAB_B	2y4gB	8	36	0.22	913	2.22787
3cbaCJ_J	3ib6A	12	23	0.52	213	2.10391
1n1iAD_D	3aqjA	12	17	0.71	257	2.36843
2dprAB_A	1yiaC	8	16	0.5	331	1.95422
3a1qDF_F	3cqxD	9	21	0.43	90	2.38014
3favAC_A	2wc8B	4	16	0.25	226	2.45435

1zaxAZ_Z	3d47G	13	17	0.76	674	2.52068
----------	-------	----	----	------	-----	---------

While not shown in the table 4.1, the algorithm could not align 5 proteins with the interface structures. The problem in the alignment part is related to the pose clustering part of algorithm where we calculated the average of the transformation parameters in a bin. A more rigorous approach to clustering algorithm can reduce the number of the transformation and align the structures better. However, the aim of QuickRet is not to align proteins but filter the database to find similar interface for the given protein in a short time. Also not shown in the table 4.1 is the 20 proteins whose interface regions are in the dataset. The algorithm correctly located the interface regions with 100% accuracy.

In the table there are 63 pairs of protein - interface structures. The numbers of the residues, proteins have, range from 32 to 2152 while the numbers of the residues in the interface structure range from 5 to 54. The average of the correctly matched residues is 0.4767 while the minimum percentage is 0.04 and the maximum percentage is 0.96 after the failed alignments and the exact alignments are removed. The histogram of the percentages is given in the graph 4.1. The average of RMSD is 2.2482 while the minimum RMSD is 0.8137 and the maximum RMSD is 3.3818. The histogram of RMSD is given in the graph 4.2. While QuickRet is not an alignment tool, the alignment it gives can be used in the next steps of PRISM. For example, TMalign accepts an initial alignment which QuickRet can provide. The average RMSD of 2.2482 is not desirable for alignment but it can work well as an initial alignment.

Below we show two case studies. There are visualization of alignments between chains and the interfaces. We also provide the list of matched residues.

1n2dAC_C & 1folA

“1n2d” is a structure of myosin V. The structure belongs to *textit{saccharomyces cerevisiae}*. Myosin V moves along actin helical track [35]. “1fol” is a structure of the

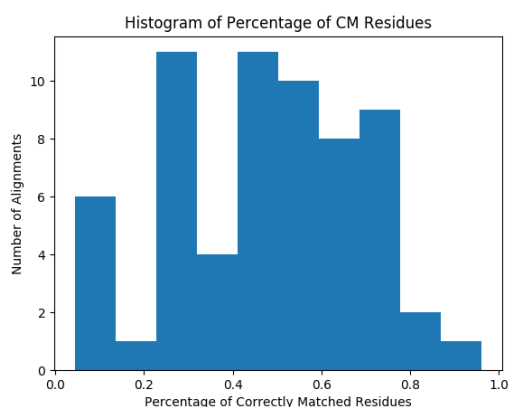


Figure 4.1: Histogram of percentage of correctly matched residues.

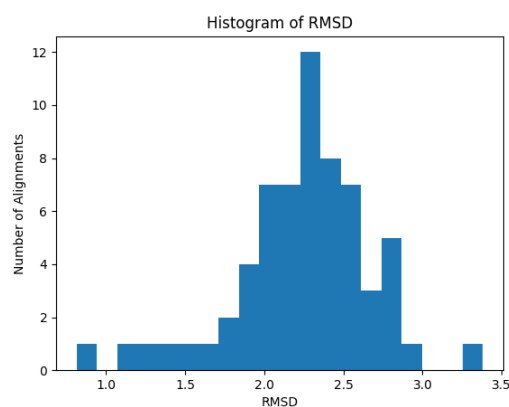


Figure 4.2: Histogram of RMSD.

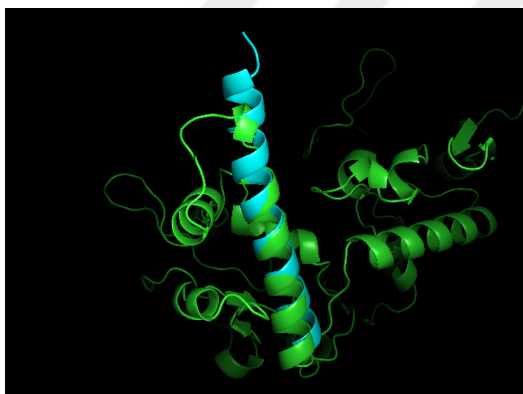


Figure 4.3: Alignment of 1n2dAC_C with 1folA.

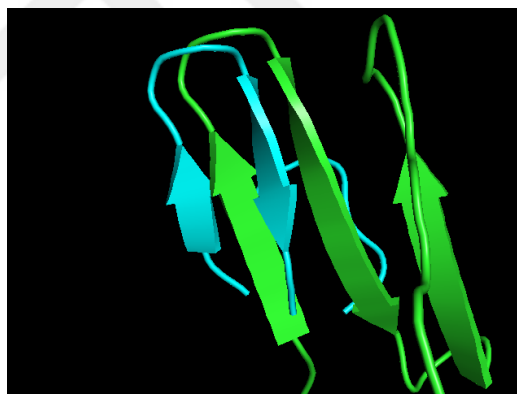


Figure 4.4: Alignment of 1n1iAD_D with 3aqjA.

endothelial nitric oxide synthase (NOS) heme domain complexed with NO [23]. The structure belongs to *textit{t}aurus*. The alignment between two structures is given in the Figure 4.3. Once again, it is a superposition of two helices. The matching pairs are given in the table 4.2. There are two pairs which have positive values according to BLOSUM62. These pairs are ILE - VAL and LYS - LYS.

1n1iAD_D & *3aqjA*

“1n1iAD” is a structure of C-terminal domains of merozoite surface protein-1. The structure belongs to *plasmodium knowlesi*. “3aqj” is a structure of a C-terminal domain of the bacteriophage P2 tail spike protein. The structure belongs to *escherichia*

Table 4.2: Matching pairs in alignment of 1n2dAC_C - 1folA

1n2dAC_C - 1folA		
812 LYS	→	393 SER
813 TYR	→	392 THR
815 GLN	→	395 LEU
816 ASN	→	396 TRP
817 ASN	→	397 LYS
818 ILE	→	398 ASP
819 LYS	→	399 LYS
820 GLY	→	400 ALA
821 PHE	→	401 ALA
822 ILE	→	402 VAL
823 ILE	→	403 GLU
824 ARG	→	404 ILE
825 GLN	→	405 ASN
826 ARG	→	406 LEU
827 VAL	→	407 ALA
828 ASN	→	408 VAL
829 ASP	→	409 LEU
830 GLU	→	410 HIS
831 MET	→	411 SER
832 LYS	→	412 PHE
833 VAL	→	413 GL
834 ASN	→	414 LEU
835 CYS	→	417 VAL

Table 4.3: Matching pairs in alignment of 1n1iAD_D - 3aqjA

1n1iAD_D - 3aqjA		
18 VAL	→	36 GLU
24 CYS	→	24 HIS
25 TYR	→	25 ILE
26 ARG	→	26 ARG
27 TYR	→	27 PHE
28 LEU	→	28 PRO
29 ASP	→	29 ASP
30 GLY	→	30 GLY
31 THR	→	31 ALA
32 GLU	→	32 VAL
33 GLU	→	33 ILE
34 TRP	→	23 LEU

phage. Quickret returns 1n1iAD_D interface structure ranked first for the protein structure of 3aqjA. The alignment of the two structures is given in the Figure 4.4. The alignment seems to be shifted which may indicate that QuickRet's pose clustering algorithm is not working properly. RMSD of the alignment is 2.37 which is worse than the average. The percentage of matched residues is 71%. Different than the previous alignment, this alignment is between β -sheets. Although, alignment is not desirable, QuickRet successfully identifies and matches secondary structures. The matching pairs between 1n1iAD_D and 1n1iAD_D is given in the table 4.3.

From the case studies presented above, we can see that there are many secondary structures in the alignments. Most of the alignments that are not shown here are between the protein structures and the helices. The small helix structures in the database can be a problem for the algorithm. Since helices are very abundant in protein structures, they can be geometrically matched with any proteins in any parts. To increase the accuracy of the algorithm, we can label helices and use these labels in the alignment part. Also, it is clear that QuickRet can geometrically find patterns that are very similar. However, it lacks biological information and the matching pairs show this lack of information.

4.1.2 Test 2: TAlign Assessments of QuickRet's results

We assessed QuickRet with an alignment tool TAlign [39]. Since TAlign is an alignment tool, the RMSD values it gives is an indicator of similarity between the interface structure and the protein structure. In the figure, we present a histogram of RMSD values calculated by TAlign for the top ranked interface structures retrieved by QuickRet. From the histogram, it is apparent that the most of the structures QuickRet retrieved are structurally very similar to the query proteins. The mean of the RMSD values is 0.85 with a standard deviation of 1.08. The maximum for the RMSD values are 3.74 which is smaller than 3.8, the threshold for alignments to be accepted as correct. Below we presented first 20 structures ranked by TAlign and QuickRet for the protein structure '1folA'. We also provided RMSD values calculated

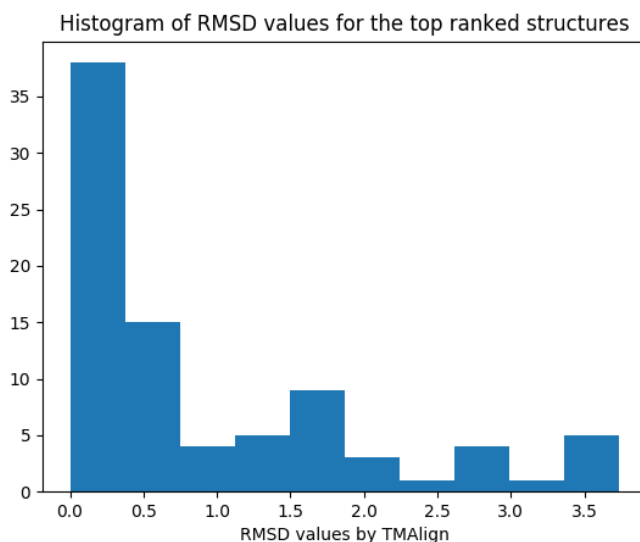


Figure 4.5: Histogram of RSMD calculated by TMalign for the top ranked structures.

by TMalign to analyze QuickRet’s results in Figure 4.5.

1n1iAD-D & *3aqjA*

1folA

While QuickRet and TMalign agrees on some alignment, there are also some interface structures alignments of which two algorithms do not agree. “1fol” is a structure of the endothelial nitric oxide synthase (NOS) heme domain complexed with NO [23]. The structure belongs to *textitbos taurus*. We showed the alignment between “1folA” and “1n2dAC_C” in the Figure 4.3. The alignment indicates that the algorithm is able to find geometrically similar patterns. TMalign also ranks “1n2dAC_C” 13th among other interfaces. TMalign ranks “1s1cBX_X” first. “1s1c” is a structure of the interaction of the human RhoA and ROCKI. The structure belongs to *homo sapiens*. They functions in important events such as cell migration, muscle contraction. It is also involved in some diseases such as asthma [9]. QuickRet ranks “1s1cBX_X” third. Also, it should be noted that the first 5 structure, except for “1n2dAC_C”, QuickRet ranks are all ranked among first 10 structures by TMaling. We present the

Table 4.4: Rankings of QuickRet and TMaling for 1folA.

1folA		
QuickRet	TMaling	
1n2dAC_C	1s1cBX_X	0.29
3favAC_C	3cbaCJ_J	0.34
1s1cBX_X	3cbaCJ_C	0.36
1zaxAZ_Z	2ahmBG_G	0.46
3a1qDF_F	1zaxAZ_Z	0.60
1hvvBC_C	3batAC_C	0.73
3cbaCJ_J	3favAC_C	0.83
3cbaCJ_C	3pdyAB_B	1.06
1hvvAB_A	3a1qDF_F	1.34
3broAD_A	3broAD_A	1.37
3pdyAB_B	1yxbBD_D	1.57
3rl0We_W	3zs9BC_C	1.66
2ahmGH_G	1n2dAC_C	1.89
3zs9BC_C	3hzeCD_D	2.00
3hzeCD_D	1hvvBC_C	2.09
2ahmBG_G	3rl0We_W	2.25
3batAC_C	2ahmGH_G	2.55
3bc1EF_F	3bc1EF_F	2.62
2efrCD_C	1hvvAB_A	3.01
1yxbBD_D	2efrCD_C	4.30

comparison between ranking in the table 4.4.

4.1.3 Test 3: PIFACE Members

We provided QuickRet proteins obtained from the clusters of PIFACE. We observed how QuickRet ranks the representatives of the clusters given a member of the cluster. We also ranked the QuickRet output with the RMSD values and the matched percentages of the interface structures calculated from the alignments QuickRet produces. We calculated the alternative scores by the equation 4.1. We subtracted the ratio from the 1.1 to favor the big interfaces which misses few residues over the small interface whose residues are completely aligned. We compare the two ranking lists for each given proteins. We conducted this test with 166 proteins. We select these proteins from the clusters which have at least 5 proteins.

$$Score = (1.1 - \frac{MatchedResidues}{InterfaceResidues}) \times RMSD \quad (4.1)$$

The test results indicates that QuickRet scoring function is not working properly compared to the alternative scoring function. While the QuickRet ranks the representatives first for the 8 given member proteins out of 166, the alternative scoring function ranks the representatives first for the 30 given member proteins out of 166. Additionally, the mean of the ranks of the representatives is 19096.95 with 11157.58 standard deviation for QuickRet while the mean is 12013.47 with 16056.27 standard deviation for the alternative scoring function. The histograms of ranks for the alternative scoring function and Quickret are given in the tables 4.6 and 4.7. In the histograms, there are 22 bins each and each bin's width is approximately 2000. It should be noted that while the alternative scoring is better overall, QuickRet ranks more representatives in first 2000 then the alternative scoring function does, 52 to 45. We present 1 case where QuickRet and alternative scoring function disagrees.

2hzyA

“2hzy” is a structure of fumarylacetoacetate hydrolase complexes. The structure belongs to *mus musculus* [2]. In the PIFACE database, the representative of the

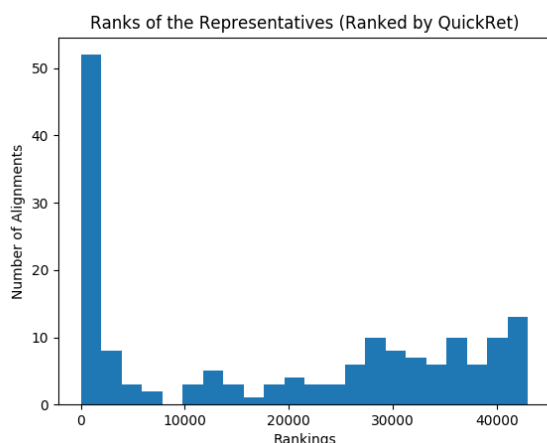


Figure 4.6: Histogram of the representatives' ranks according to QuickRet.

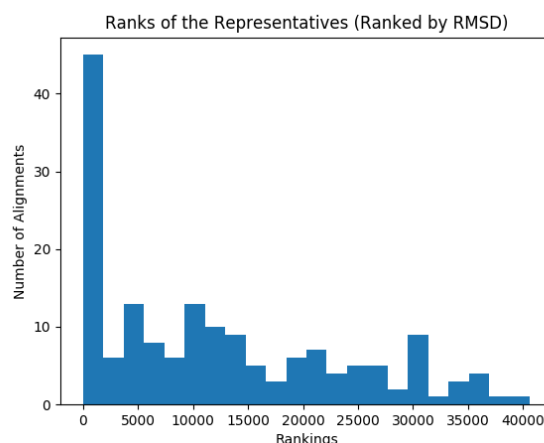


Figure 4.7: Histogram of the representatives' ranks according to the alternative scoring function.

group to which “2hzyAB” belongs is the structure “1qqjAB” which also is a structure of fumarylacetoacetate hydrolase complexes which belongs to *mus musculus* [36]. QuickRet ranks the structure “2wojAD_A” first for the A chains of “2hzy.” “2woj” is a structure of ADP-AIF4 complex. The structure belongs to *saccharomyces cerevisiae* [26]. The alternative scoring function ranks the representative of the group, which is “1qqjAB_A,” first. The alignments of the structure “2hzyA” with “1qqjAB_A” and “2wojAD_A” are shown in the Figures 4.8 and 4.9. We also provide structures of the interfaces in the Figures 4.10 and 4.11.

From the figures, it is very apparent that the alternative scoring function does a better job at finding a similar structure for the given structure “2hzy.” While “1qqjAB_A” is the representative of the group, “2hzyAB” belongs to, the alignment between them matches all of the 111 residues of the structure “1qqjAB_A” with the structure “2hzyA.” The RMSD value of the alignment is 0.183 which is a very good score. On the other hand, QuickRet ranks “2wojAD_A”, a small helix structure, first. The alignment between two structures, as seen in the Figure 4.9, does not have a good superpositioning of the interface structure onto the protein structure. The RMSD value of the alignment is 2.342. The RMSD value should be lower, however the algorithm matches the closest pairs together. The alignment matches all 8 residues

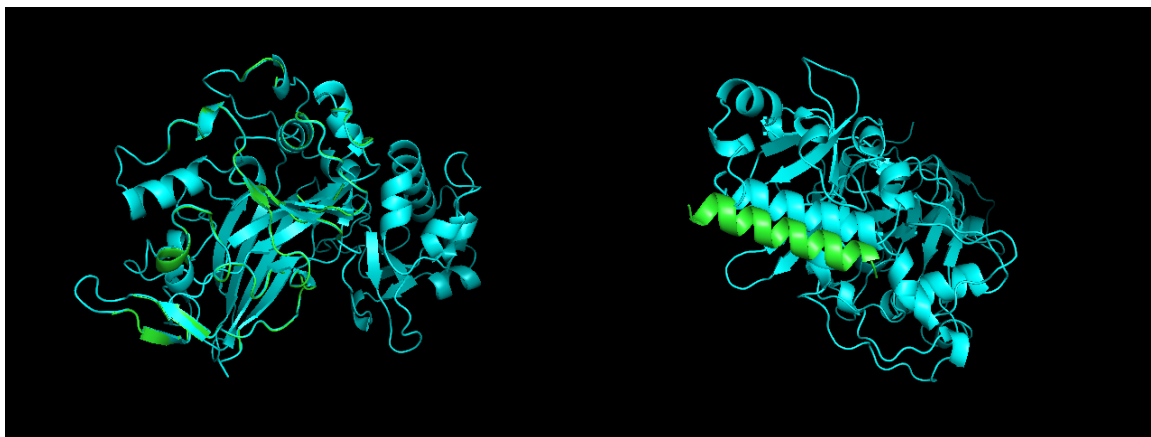


Figure 4.8: Alignment of 1qqjAB_A with 2hzyA. Figure 4.9: Alignment of 2wojAD_A with 2hzyA.

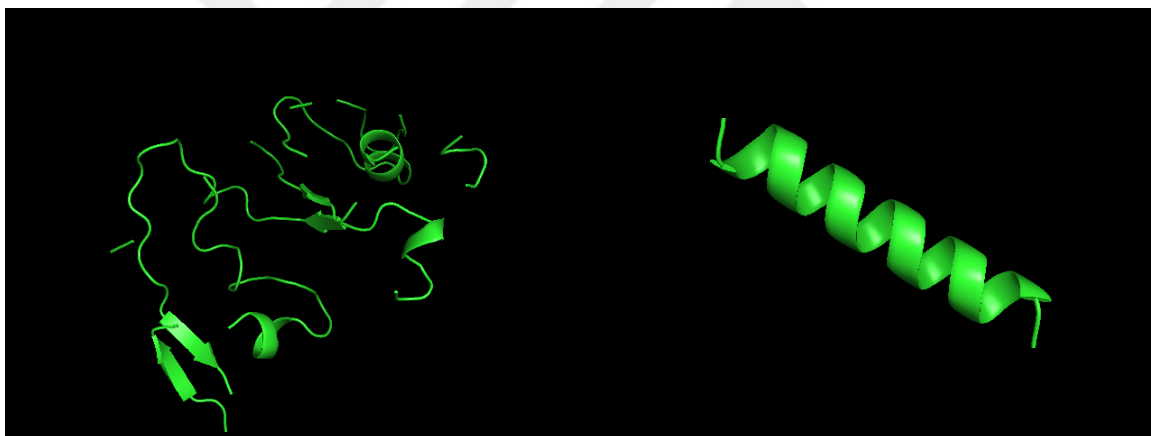


Figure 4.10: Structure of 1qqjAB_A.

Figure 4.11: Structure of 2wojAD_A.

of the interface structure which has 20 residues.

The histograms in the Figures 4.6 and 4.7 and the case study clearly show that the scoring function of QuickRet does not properly ranked the interface structures. It favors the small helices to more meaningful structures which are bigger.

4.1.4 Test 4: PRISM Predictions

We also tested QuickRet with 80 predictions made by PRISM. PRISM docks two given protein with the template interfaces in the database. Both sides of the interface, PRISM chooses to use as a template should be similar to the given proteins. We expect QuickRet to rank a side of the template interface predicted by PRISM higher in the list, when the protein, PRISM predicted for, is given. We used 80 pairs extracted

from PRISM predictions in this test. The mean of the energy values calculated by PRISM for the prediction is -31.97 with the standard deviation of 62.60.

The histograms for the ranks of the template interfaces are given in the Figures 4.16 and 4.17. Once again, we also compare the QuickRet's ranking with the alternative scoring function's ranking which is given in the equation 4.1. The alternative scoring function produces a better ranking overall compared to Quickret's. The mean of the alternative scoring function's ranks for the template interfaces is 15500.96 with the standard deviation of 8818.86 while the mean of the QuickRet's ranks for the template interfaces is 23228.91 with the standard deviation of 13644.77. However, QuickRet ranks 13 template interfaces in the first 5000 while the alternative scoring function ranks 11 template interfaces in the first 5000. We also presented 2 case studies in one of which the alternative scoring function and QuickRet's scoring function agree, and in other the two scoring function disagree.

Additionally, we filtered the results for the 80 proteins according to the percentage of contact residue match to interface residues and the percentage of match residues to interface residues to see if QuickRet's results are meaningful. The results are given in the Tables A.1 and A.2. As the thresholds for the matched percentages increase, if the single matches resulted in smaller RMSD, RMSD values should increase. However, RMSD values decreases as thresholds increase. Thus, we claim that QuickRet is able to find meaningful interface structure even though it may miss PRISM predictions. Also, the percentage contact residues match filters hasher than the other filter. It is not surprising since contact residues constitute a smaller set. However, for the first threshold (0.1), we see that number of remaining structures is maximum 21000 and minimum 10000. The result indicates that QuickRet matches most of the interfaces to surfaces of proteins. It is important because it eliminates the additional step of surface extraction which saves so much time. Finally the trends of the average RMSD values and the average of remaining interface structures are given in Figures 4.12, 4.13, 4.14 and 4.15. The average of remaining interfaces decreases as the average RMSD values decreases which is a good indication that QuickRet filters unrelated structures

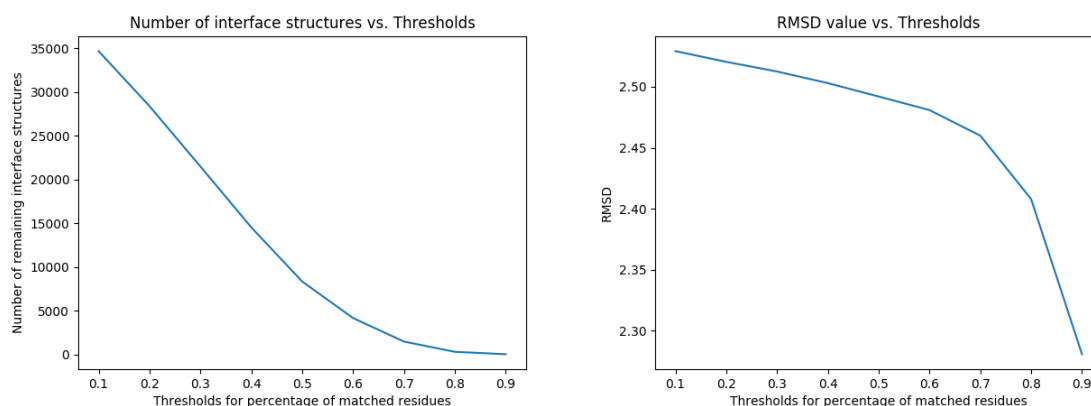


Figure 4.12: Graph of the average number of re-
Figure 4.13: Graph of the average RMSD value
maining interface structures for given percentageof remaining interface structures with the given
protein for given percentage match thresholds.

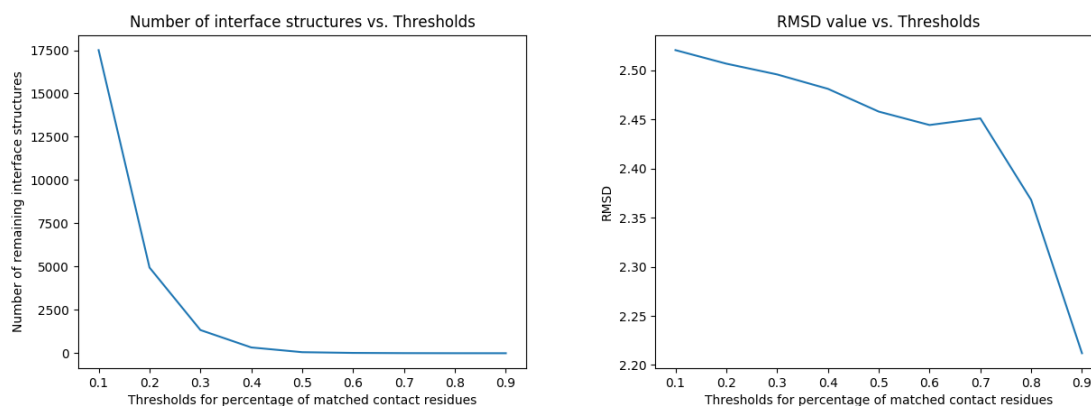


Figure 4.14: Graph of the average number of re-
Figure 4.15: Graph of the average RMSD value
maining interface structures for given percentageof remaining interface structures with the given
protein for given percentage contact match thresh-
olds.

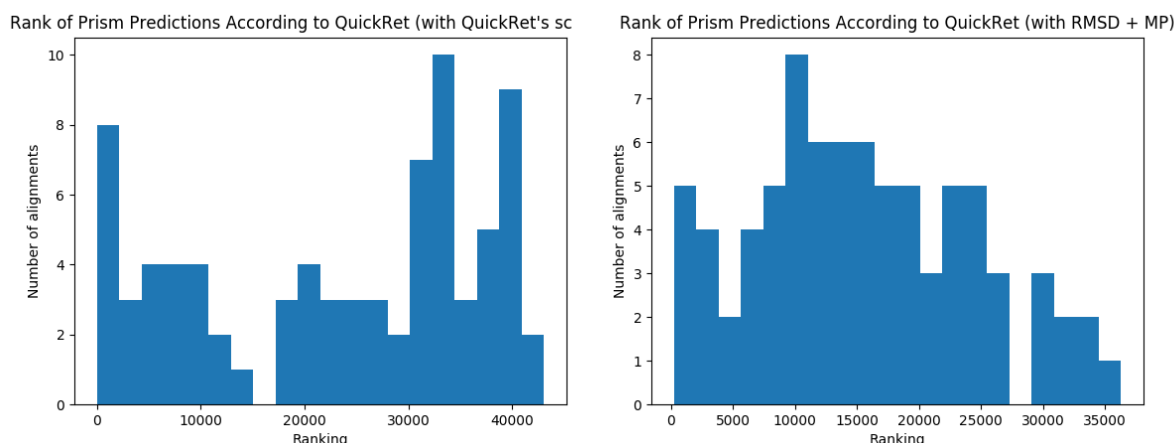


Figure 4.16: Histogram of the templates' ranks according to QuickRet. Figure 4.17: Histogram of the templates' ranks according to the alternative scoring function.

while it keeps related ones which can be aligned with the given protein with desirable RMSD values.

1tupB

“1tup” is a structure of tumor suppressor P53 complexed with DNA. It belongs to *homo sapiens* [6]. Both QuickRet and the alternative scoring function agree on the best interface structure for the “1tupB” which is “1tsrAB_B.” “1tsr” is a structure of P53 core domain in complex with DNA. It also belongs to *homo sapiens* [6]. The RMSD value for the alignment between “1tupB” and “1tsrAB_B” is 0.03. Quickret aligns all of the 37 residues of the template interface to the protein correctly. “1tsrAB” is also the representative of the cluster to which “1tupAB” belongs to. The RMSD value, the percentage match and the PIFACE cluster information all point out that the alignment is correct. The alignment is given in the Figure 4.18 and the structure of “1tsrAB_B” is given in the Figure 4.19.

1nvwR

“1nvw” is a structure of RasGTP. The structure belongs to *homo sapiens* [25]. The alternative scoring function ranks “1nvuRS_R” first which is also a structure of RasGTP which belongs to *homo sapiens* [25]. The RMSD value for the alignment of

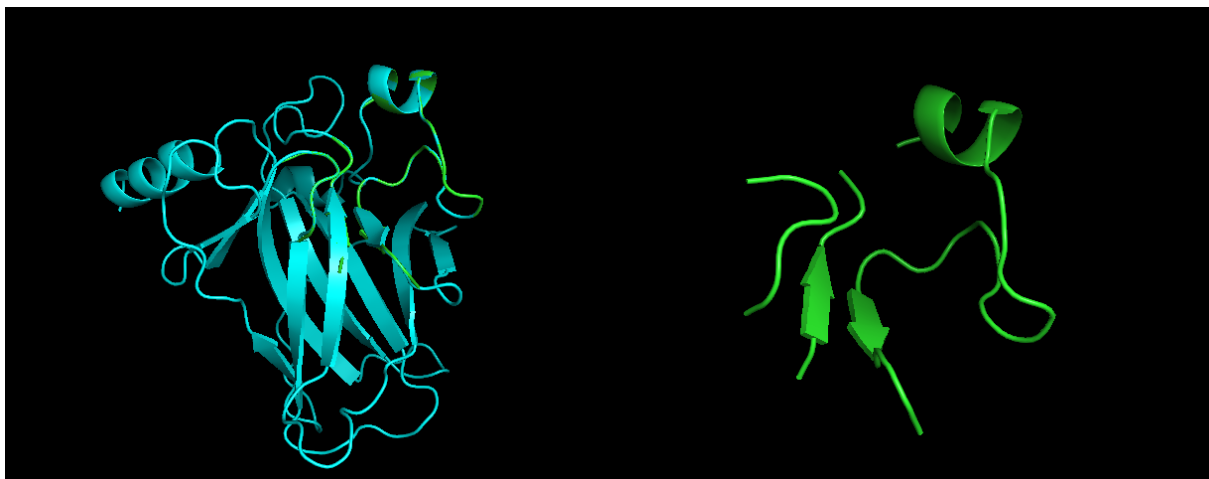


Figure 4.18: Alignment of 1tsrAB_B with 1tupB.

Figure 4.19: Structure of 1tsrAB_B.

the two structure is 0.35. QuickRet manages to match all the 69 residues of the template interface with the query protein. Also, the representative of the cluster to which “1nvwRS” is “1nvuRS.” All the information on the alignment indicates that the alignment is correct. The alignment is given in the Figure 4.20 and the interface structure “1nvuRS_R” is shown in the Figure 4.22. While “1nvuRS_R” seems to be the correct match, QuickRet ranks “2f4iAB_B” first. “2f4i” is a structure of an ob-fold protein. It belongs to *thermotoga maritima*. The alignment between the two structures is 2.35 and QuickRet manages to align 9 residues of the template interface which has 14 residues. The interface structure and the alignment are given in the Figure 4.21 and the Figure 4.23 consecutively. We can see that the QuickRet’s scoring functions favors the smaller helix over the more meaningful, bigger interface structure.

4.1.5 Test 5: Algorithm Speed

The best feature of QuickRet is the ability to quickly scan the database of interfaces and rank similar structures to the query protein. In this study, we tested the speed of the algorithm. We compared the speed of the QuickRet with TMalign on whole interface database. Also we observed the affect of the threshold applied to the bond angles, the torsion angle and the distance between C_{α} . We increased the threshold by

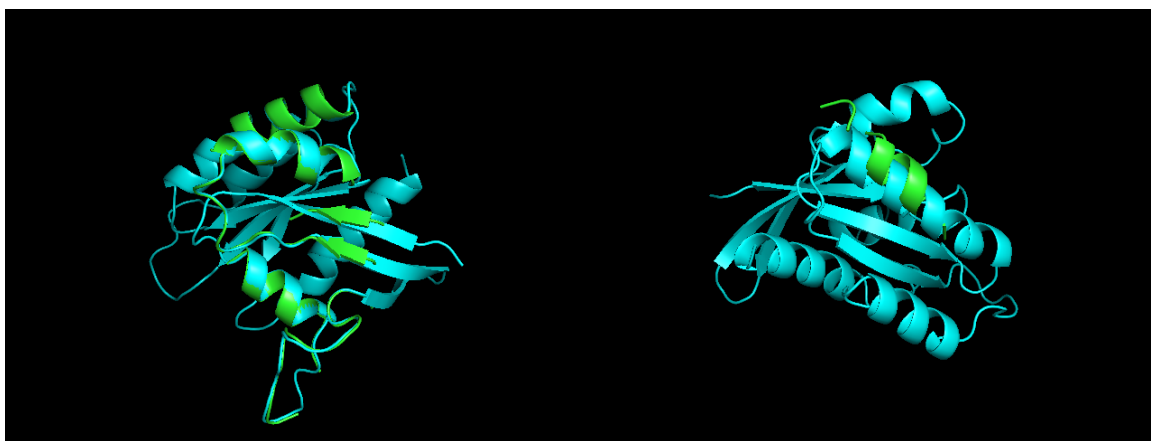


Figure 4.20: Alignment of 1nvuRS_R with 1nvwR. Figure 4.21: Alignment of 2f4iAB_B with 1nvwR.

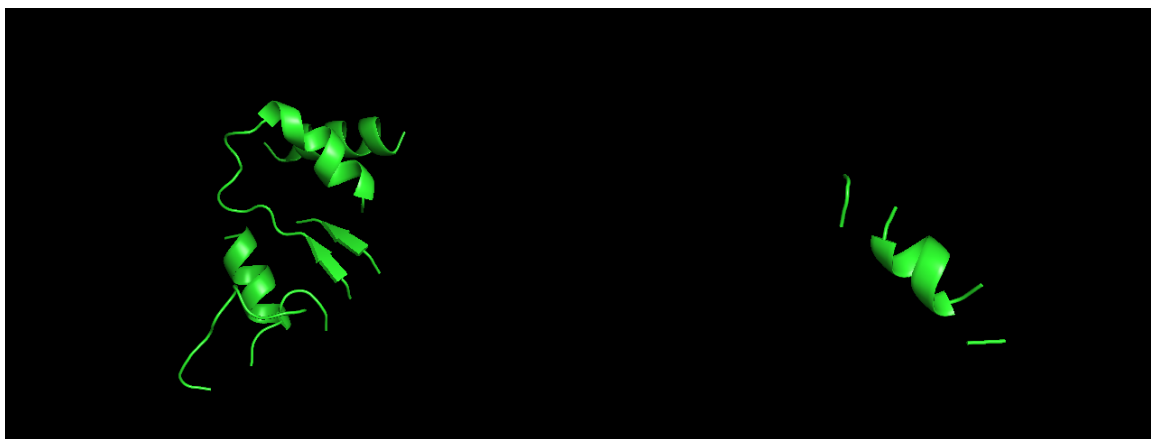


Figure 4.22: Structure of 1nvuRS_R.

Figure 4.23: Structure of 2f4iAB_B.

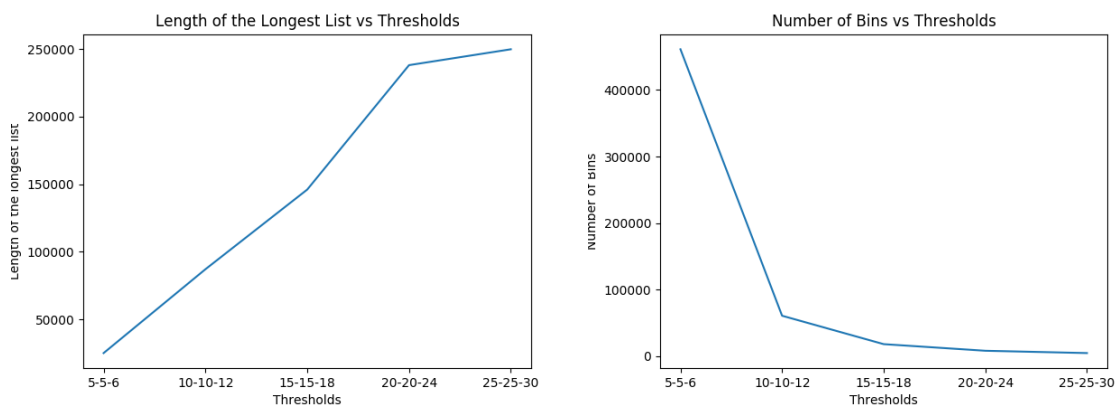


Figure 4.24: Graph of the length of the longest list in the hash table vs thresholds. Figure 4.25: Graph of the number of bins in the hash table vs thresholds.

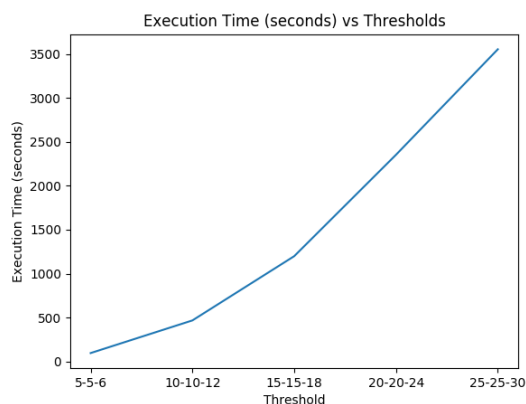


Figure 4.26: Graph of the execution time vs thresholds.

5° for the bond angles and the torsion angles and $6 \text{ \AA}$ for the distance between two C_α in every step. We set the initial value for the threshold to 5° for the bond angles and the torsion angle, and $6 \text{ \AA}$ for the distance. We use these thresholds for the accuracy tests since they are the best performing. We increased the thresholds until the final values of 25° for the bond angles, and the torsion angle and $30 \text{ \AA}$.

As we increase the thresholds, we observed that the accuracy decreases steeply. The speed of the algorithm also dramatically decreases. We can identify the reason behind the reduction of speed and accuracy by analyzing the number of the bins in the hash table and the number of the entries in the bins. In the Figure 4.24, we present the change in the length of the longest list as the thresholds increase. Since

Table 4.5: Execution time of TMalign and QuickRet.

TMalign	QuickRet
16m37.847s	1m35.429s

the lookup complexity for the list data structure is $\mathcal{O}(n)$ as the lists get longer, the algorithm's speed dramatically decreases. In the Figure 4.24, we can see a steep increase in the length of the lists as the thresholds increase. The steep increase in the length of the lists is natural since as the thresholds increase, algorithm clumps more features in the same bin. We also presented the graph of change in number of bins as the thresholds increase in the Figure 4.25. The two graphs agrees on the effect of an increase in the thresholds. As the thresholds increase, the number of bins decreases since the algorithm clusters the features into less bins. This effects the accuracy of the algorithm, since it votes every element of a bin when a feature points to the bin. We also show the effect of a change in thresholds on the execution time of the algorithm for the structure "3mvkA" in the Figure 4.26. The graph conforms with the previous graphs. Since the length of lists increases, the execution time increases proportionally as expected.

Finally we compare the execution times of QuickRet and TMalign. We aligned each interface in the data base with "3mvkA" and sorted them according to the RMSD values. The execution time of the two algorithm is listed in the table 4.5. While QuickRet ranks the structures in 1 minutes and 30 seconds, TMalign ranks them in 16 minutes and 37 seconds. QuickRet is approximately 10 times faster than TMalign. A user can choose to wait for 16 minutes for a better alignment however as the size of the data set grows, the benefits of QuickRet become more obvious. For example for a data set of 1000000 interfaces, the scaled execution times of the algorithms are 377 minutes 39 seconds for TMalign and 35 minutes 59 seconds for QuickRet. Since adding more interfaces to the data set increases accuracy, the system benefits from QuickRet to reduce the execution time drastically. Adding biological information to the hashing algorithm of TMalign should not change the complexity

of the algorithm because hashing additional biological data has constant complexity. Thus even if QuickRet is modified, the speed advantage of the algorithm against TAlign should not change dramatically.

4.2 Interface Prediction with Deep Learning Methods

We initially tried a small subset of the data which consists of approximately 20000 samples for training and 5000 samples for testing. The initial dataset for training and testing the model includes only DOCKGROUND as negative interface source and PIFACE as positive interface source. The result of the initial model which is similar to Le-Net achieved a test accuracy of 90%. However, when we tried the model with the latest dataset which is described above in the method section, the accuracy of the model decreased dramatically to 67%. Drop in the accuracy required us to come up with a more complex and deeper model. We created a deeper model whose accuracy is given in the table 4.6. The first result in the table 4.6 is obtained by testing the model with the dataset including samples only from PIFACE and PPI4DOCK. The second result in the table 4.6 is obtained by testing the model with the training dataset including samples from PIFACE and PPI4DOCK and the testing dataset including samples from DOCKGROUND and the structures registered to PDB after 2012.

Since “negative” interface concept is nebulous, creating a negative dataset is a difficult task. First of all, an interface labeled as ‘negative’ can be very similar to an interface of a structure functioning in a very different environment. This makes it very difficult for a model to differentiate between positive and negative interfaces. One solution to this problem can be providing additional information about the environment the protein functions such as pH. The second problem is to create a negative dataset which covers most of the negative interface space. Since there are many possibilities to dock 2 proteins in 3 dimensions and very few of these docked structures are correct, there are so many possibilities to consider when constructing a negative dataset for interface structures.

We also analyzed the composition of the interfaces coming from different sources.

Table 4.6: Results of the second model for the two datasets.

	positive	negative
Tr. and test from the same source	88.76%	96.12%
Tr. and test from the different sources	83.64%	73.24%

We grouped the amino acids into 7 groups according to their chemical properties. These groups are; aliphatic group, hydroxyl group, sulfur group, acidic group, basic group, aromatic group, imino group. We classified GLY, ALA, VAL, LEU, and ILE into the aliphatic group; SER, THR, TYR into the hydroxyl group; CYS, MET into the sulfur group; ADP, ASN, GLU, and GLN into the acidic group; LYS, ARG, HIS into the basic group; PHE, TYR, TRP Into the aromatic group; and finally, PRO into the imino group. We derived the distribution of percentages of the group in interfaces structures for the each source. We compare this distributions derived from different source with each other. The comparisons for each groups are given in the Figures 4.27, 4.28, 4.29, 4.30, 4.31, 4.32, and 4.33. The p-value test shows us that the composition of the interfaces for the negative datasets are very different. This is definitely a part of the problem. However, there is also structural considerations which we mentioned in the previous paragraph.

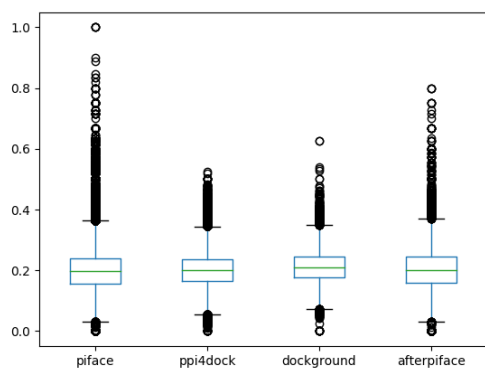


Figure 4.27: Distributions of acidic composition percentages.

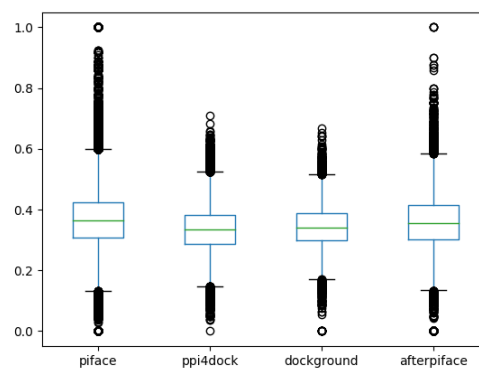


Figure 4.28: Distributions of aliphatic composition percentages.

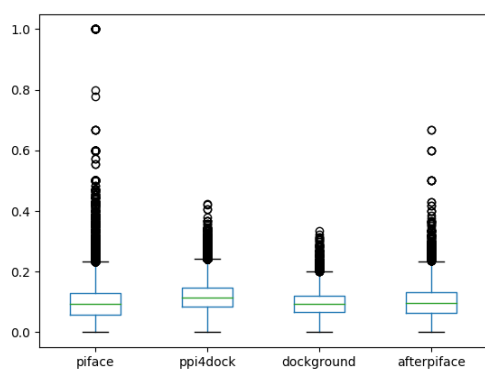


Figure 4.29: Distributions of aromatic composition percentages.

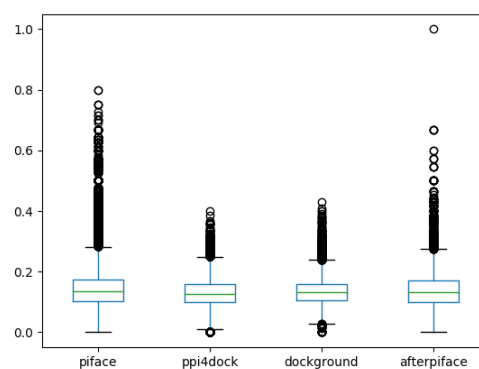


Figure 4.30: Distributions of basic composition percentages.

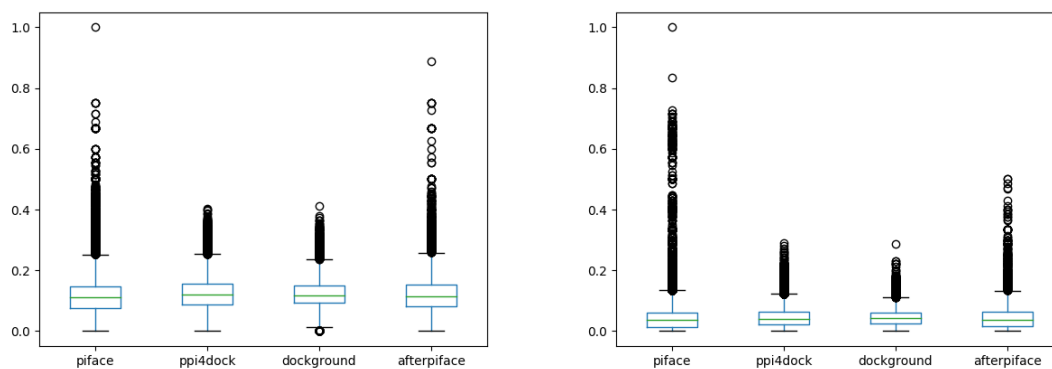


Figure 4.31: Distributions of hydroxyl composition percentages. Figure 4.32: Distributions of imino composition percentages.

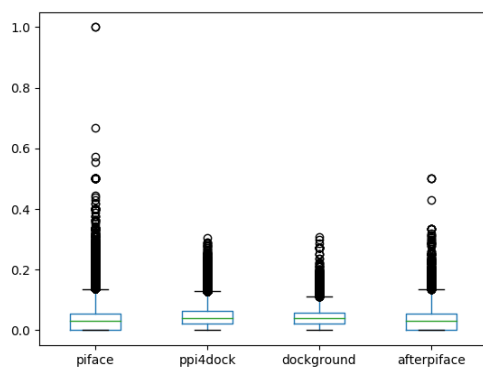


Figure 4.33: Distributions of sulfur composition percentages.

Chapter 5

CONCLUSION

Obtaining information on protein - protein interactions is very important. However, experimentally doing so takes so much time and effort. Computational tools are very fast and getting more reliable as more research is done. Since discovering new protein - protein interactions are very important, more research should be conducted to develop reliable and fast tools to discover new protein - protein interactions. In this thesis, we aimed at developing two different tools for two different steps of the discovery process of the protein - protein interactions. The first tool we developed, QuickRet, quickly scans through a database of interfaces and returns the most similar one for the given protein. The second tool we developed is a convolutional neural network which gives the possibility of a docked complex being valid.

We showed that QuickRet is 10 times faster than scanning a database of interfaces with an alignment tool while it produces reliable results. However, the scoring functions which is used to rank results needs an improvement. We show that the scoring function favors smaller but not very meaningful results for some of the structure. Ranking results with RMSD values and the percentage of interface which is matched by QuickRet showed that the algorithm is able to find meaningful results. With RMSD values and the percentage match, QuickRet is able to reduce the set of interfaces by filtering dissimilar structures. We are confident that if the scoring function is improved, the algorithm will become a reliable tool for the discovery of new protein - protein interactions.

We showed that the neural network we trained with interface structures is able to learn features relevant to identify interface structures. However, it fails to generalize for negative interfaces. We showed that the negative databases are very different from

each other which makes training networks to identify negative interfaces very difficult. More research should be done on creating negative protein protein interaction databases.

With the suggested improvements done, a fast and reliable tool for the discovery of protein - protein interaction tool can be implemented. Implementation of such a tool will lead to discovery of many protein - protein interaction which will affect many biological sciences positively.

BIBLIOGRAPHY

- [1] Baspinar, A., Cukuroglu, E., Nussinov, R., Keskin, O., and Gursoy, A. (2014). Prism: a web server and repository for prediction of protein–protein interactions and modeling their 3d complexes. *Nucleic acids research*, 42(W1):W285–W289.
- [2] Bateman, R. L., Ashworth, J., Witte, J. F., Baker, L.-J., Bhanumoorthy, P., Timm, D. E., Hurley, T. D., Grompe, M., and Mcclard, R. W. (2007). Slow-onset inhibition of fumarylacetoacetate hydrolase by phosphinate mimics of the tetrahedral intermediate: kinetics, crystal structure and pharmacokinetics. *Biochemical Journal*, 402(2):251–260.
- [3] Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. (2006). The protein data bank, 1999–. In *International Tables for Crystallography Volume F: Crystallography of biological macromolecules*, pages 675–684. Springer.
- [4] Chen, R., Li, L., and Weng, Z. (2003). Zdock: an initial-stage protein-docking algorithm. *Proteins: Structure, Function, and Bioinformatics*, 52(1):80–87.
- [5] Chen, R. and Weng, Z. (2003). A novel shape complementarity scoring function for protein-protein docking. *Proteins: Structure, Function, and Bioinformatics*, 51(3):397–408.
- [6] Cho, Y., Gorina, S., Jeffrey, P. D., and Pavletich, N. P. (1994). Crystal structure of a p53 tumor suppressor-dna complex: understanding tumorigenic mutations. *Science*, 265(5170):346–355.
- [7] Cukuroglu, E., Gursoy, A., Nussinov, R., and Keskin, O. (2014). Non-redundant unique interface structures as templates for modeling protein interactions. *PloS one*, 9(1):e86738.

- [8] Day, M. and Games, I. (2012). Extracting euler angles from a rotation matrix. *Insomniac Games R&D*. Available online at: <http://www.insomniacgames.com/mike-day-extracting-euler-angles-from-a-rotation-matrix>.
- [9] Dvorsky, R., Blumenstein, L., Vetter, I. R., and Ahmadian, M. R. (2004). Structural insights into the interaction of rocki with the switch regions of rhoa. *Journal of Biological Chemistry*, 279(8):7098–7104.
- [10] Estrin, M. and Wolfson, H. J. (2017). Snapdocktemplate-based docking by geometric hashing. *Bioinformatics*, 33(14):i30–i36.
- [11] Girvan, M., Callaway, D. S., Newman, M. E., and Strogatz, S. H. (2002). Simple model of epidemics with pathogen mutation. *Physical Review E*, 65(3):031915.
- [12] Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256.
- [13] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.
- [14] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. pages 770–778.
- [15] Hubbard, S., Campbell, S., and Thornton, J. (1991). Molecular recognition: conformational analysis of limited proteolytic sites and serine proteinase protein inhibitors. *Journal of molecular biology*, 220(2):507–530.
- [16] Janin, J., Henrick, K., Moult, J., Eyck, L. T., Sternberg, M. J., Vajda, S., Vakser, I., and Wodak, S. J. (2003). Capri: a critical assessment of predicted interactions. *Proteins: Structure, Function, and Bioinformatics*, 52(1):2–9.

- [17] Kabsch, W. (1976). A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923.
- [18] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- [19] Kufareva, I. and Abagyan, R. (2011). Methods of protein structure comparison. In *Homology Modeling*, pages 231–257. Springer.
- [20] Lamdan, Y. and Wolfson, H. J. (1988). Geometric hashing: A general and efficient model-based recognition scheme.
- [21] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998a). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- [22] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998b). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- [23] Li, H., Raman, C., Martásek, P., Masters, B. S. S., and Poulos, T. L. (2001). Crystallographic studies on endothelial nitric oxide synthase complexed with nitric oxide and mechanism-based inhibitors. *Biochemistry*, 40(18):5399–5406.
- [24] Liu, S., Gao, Y., and Vakser, I. A. (2008). Dockground protein–protein docking decoy set. *Bioinformatics*, 24(22):2634–2635.
- [25] Margarit, S. M., Sondermann, H., Hall, B. E., Nagar, B., Hoelz, A., Pirruccello, M., Bar-Sagi, D., and Kuriyan, J. (2003). Structural evidence for feedback activation by ras·gtp of the ras-specific nucleotide exchange factor sos. *Cell*, 112(5):685–695.

- [26] Mateja, A., Szlachcic, A., Downing, M. E., Dobosz, M., Mariappan, M., Hegde, R. S., and Keenan, R. J. (2009). The structural basis of tail-anchored membrane protein recognition by get3. *Nature*, 461(7262):361.
- [27] Nussinov, R. and Wolfson, H. J. (1991). Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques. *Proceedings of the National Academy of Sciences*, 88(23):10495–10499.
- [28] Particle Sciences (2009). Levels of protein structure. [Online; accessed July 24, 2018].
- [29] Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.
- [30] Russell, P. J. and Gordey, K. (2002). *IGenetics*. Number QH430 R87. Benjamin Cummings San Francisco.
- [31] Shatsky, M., Nussinov, R., and Wolfson, H. J. (2004). A method for simultaneous alignment of multiple protein structures. *Proteins: Structure, Function, and Bioinformatics*, 56(1):143–156.
- [32] Söding, J. (2004). Protein homology detection by hmm–hmm comparison. *Bioinformatics*, 21(7):951–960.
- [33] Stockman, G. (1987). Object recognition and localization via pose clustering. *Computer vision, graphics, and image processing*, 40(3):361–387.
- [34] Sun, T., Zhou, B., Lai, L., and Pei, J. (2017). Sequence-based prediction of protein protein interaction using a deep-learning algorithm. *BMC bioinformatics*, 18(1):277.
- [35] Terrak, M., Rebowski, G., Lu, R. C., Grabarek, Z., and Dominguez, R. (2005). Structure of the light chain-binding domain of myosin v. *Proceedings of the National Academy of Sciences*, 102(36):12718–12723.

- [36] Timm, D. E., Mueller, H. A., Bhanumoorthy, P., Harp, J. M., and Bunick, G. J. (1999). Crystal structure and mechanism of a carbon-carbon bond hydrolase. *Structure*, 7(9):1023–1033.
- [37] Tovchigrechko, A. and Vakser, I. A. (2006). Gramm-x public web server for protein-protein docking. *Nucleic acids research*, 34(suppl_2):W310–W314.
- [38] Yu, J. and Guerois, R. (2016). Ppi4dock: large scale assessment of the use of homology models in free docking over more than 1000 realistic targets. *Bioinformatics*, 32(24):3760–3767.
- [39] Zhang, Y. and Skolnick, J. (2005). Tm-align: a protein structure alignment algorithm based on the tm-score. *Nucleic acids research*, 33(7):2302–2309.
- [40] Zhu, H., Domingues, F. S., Sommer, I., and Lengauer, T. (2006). Noxclass: prediction of protein-protein interaction types. *BMC bioinformatics*, 7(1):27.



Appendices

Appendix 1

APPENDIX A

Table A.1: Filtering interface structures according to percentage match.

#int: Number of interface structures which passes the filter M_{RMSD} : Average RMSD

σ_{RMSD} : standard deviation for RMSD

Protien	> 0.1			> 0.2			> 0.3		
	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}
2evaA	38125	2.54	0.23	33010	2.54	0.22	26458	2.53	0.21
3oe8C	37658	2.53	0.25	31370	2.52	0.23	23527	2.51	0.23
1oedB	31877	2.51	0.27	23629	2.5	0.25	15423	2.49	0.25
5l7gA	37565	2.53	0.24	32411	2.53	0.23	25756	2.53	0.22
1aifB	34823	2.54	0.24	27881	2.54	0.22	19619	2.53	0.21
1i4eB	37209	2.53	0.23	31813	2.53	0.21	25029	2.52	0.2
1oedA	31743	2.52	0.27	23385	2.5	0.26	14943	2.49	0.25
2ghuA	38207	2.53	0.21	33766	2.52	0.2	28163	2.52	0.19
5l7eA	37608	2.53	0.24	32482	2.53	0.23	25905	2.52	0.22
4n8eB	40108	2.54	0.22	36117	2.54	0.21	30253	2.53	0.2
1blxA	38456	2.55	0.23	33189	2.54	0.22	26275	2.54	0.21
3a20A	33306	2.52	0.24	26184	2.52	0.23	18529	2.51	0.22
3udwB	29191	2.53	0.23	22734	2.52	0.21	15709	2.52	0.2
4lgdG	17964	2.47	0.34	7621	2.42	0.33	2746	2.37	0.34
3o47B	36759	2.53	0.24	30710	2.53	0.22	23366	2.52	0.22
2y83S	39176	2.53	0.22	34892	2.53	0.2	29170	2.52	0.19

2y83R	39047	2.53	0.22	34542	2.53	0.2	28697	2.52	0.2
3omvB	37801	2.55	0.23	32669	2.54	0.21	26018	2.54	0.21
5fd2A	37395	2.54	0.24	31988	2.54	0.22	25309	2.53	0.21
1yhwA	37921	2.53	0.23	32639	2.52	0.22	26039	2.52	0.21
1aboA	21403	2.52	0.24	14886	2.51	0.22	9104	2.5	0.21
4qo1A	32373	2.53	0.23	26285	2.52	0.21	19481	2.52	0.2
4tq9B	35354	2.53	0.24	28916	2.52	0.22	22004	2.52	0.21
1s9iA	38186	2.54	0.23	33179	2.54	0.21	26955	2.54	0.21
1g50A	37113	2.53	0.24	31489	2.53	0.23	24510	2.52	0.22
4tntA	29105	2.53	0.25	21483	2.52	0.23	14154	2.51	0.23
1g50C	37131	2.53	0.24	31519	2.53	0.23	24584	2.52	0.22
2k7zA	36663	2.57	0.23	30558	2.56	0.22	22686	2.56	0.21
1ap2B	31803	2.53	0.23	25592	2.52	0.21	18643	2.52	0.2
1m2zA	37819	2.54	0.24	32829	2.53	0.22	26269	2.53	0.22
1a2vC	38652	2.54	0.23	33570	2.53	0.22	27402	2.53	0.21
1tupB	35719	2.53	0.23	30009	2.53	0.21	23416	2.52	0.2
1jaiA	35659	2.53	0.24	29548	2.53	0.22	22559	2.52	0.21
1bmfE	39055	2.53	0.22	34472	2.52	0.21	28322	2.52	0.2
6q21B	35967	2.53	0.23	30151	2.52	0.22	23265	2.52	0.21
4obeA	35237	2.53	0.24	28885	2.52	0.23	21901	2.52	0.22
1dykA	38222	2.52	0.22	33652	2.52	0.2	27375	2.51	0.19
1qupA	36004	2.53	0.24	29660	2.53	0.22	22122	2.52	0.21
1nobF	35473	2.53	0.22	29893	2.52	0.21	22964	2.52	0.2
4dsoA	35879	2.53	0.24	29420	2.52	0.23	22248	2.52	0.22
4eklA	38460	2.54	0.23	33651	2.54	0.21	27325	2.53	0.2
5hheA	18181	2.45	0.37	7985	2.38	0.37	3106	2.3	0.41
1nvwR	35490	2.54	0.24	29148	2.53	0.22	22018	2.52	0.22
4csjA	37350	2.54	0.24	31776	2.54	0.22	24890	2.53	0.22
1g5mA	35661	2.54	0.23	29670	2.54	0.22	22481	2.53	0.21

3ldqB	30726	2.52	0.27	22173	2.5	0.26	13987	2.49	0.26
2pqfA	35746	2.53	0.23	29704	2.53	0.22	22528	2.52	0.21
4lqyA	38864	2.53	0.22	34394	2.53	0.21	28499	2.52	0.2
1he8B	35501	2.53	0.24	29325	2.52	0.22	22551	2.51	0.21
4p6xC	37704	2.54	0.24	32376	2.53	0.23	25534	2.53	0.22
5l7hA	37705	2.54	0.24	32336	2.53	0.23	25608	2.52	0.22
1yjdC	31605	2.53	0.23	25496	2.52	0.21	18442	2.51	0.2
3jckG	33212	2.53	0.27	23544	2.51	0.25	14799	2.51	0.24
1xd2C	38594	2.54	0.25	33094	2.53	0.23	25883	2.53	0.23
4udcA	37599	2.54	0.24	32385	2.53	0.22	25610	2.53	0.22
3szaB	38140	2.53	0.24	32929	2.53	0.22	26530	2.52	0.21
4p6xA	37903	2.54	0.24	32744	2.53	0.22	26005	2.53	0.22
1hurA	36256	2.54	0.23	30283	2.53	0.22	23334	2.53	0.21
4jupB	29835	2.53	0.26	21916	2.52	0.24	14381	2.51	0.23
1na7A	39121	2.55	0.22	34711	2.54	0.21	28507	2.54	0.2
7catB	38298	2.54	0.24	32412	2.53	0.22	25386	2.53	0.21
1qduA	33485	2.53	0.25	26141	2.52	0.23	18460	2.51	0.22
3mupA	30313	2.53	0.26	22110	2.52	0.23	14600	2.52	0.22
2m1wA	34194	2.53	0.25	26879	2.53	0.23	19269	2.52	0.22
4j93A	33902	2.52	0.26	26936	2.51	0.24	19273	2.5	0.23
4pf3A	37497	2.53	0.24	32331	2.52	0.23	25506	2.52	0.23
3zzxB	32347	2.52	0.26	24813	2.51	0.24	17277	2.51	0.23
2kecA	25100	2.52	0.28	15610	2.51	0.26	8671	2.5	0.25
6q21D	36043	2.53	0.24	30033	2.52	0.22	23073	2.52	0.21
2aa2A	37412	2.53	0.25	31952	2.52	0.23	25199	2.52	0.23
2jcwA	33437	2.51	0.23	27148	2.5	0.21	20505	2.5	0.2
1r6qA	34056	2.52	0.25	27227	2.51	0.24	19901	2.5	0.23
2geqA	34649	2.54	0.25	27316	2.53	0.23	19784	2.52	0.21
1e96A	35359	2.53	0.24	29045	2.53	0.22	22132	2.52	0.21

5g3jA	37580	2.54	0.24	32097	2.53	0.23	25308	2.53	0.22
4jnkA	38050	2.53	0.24	32875	2.52	0.22	26677	2.51	0.21
1b34A	27148	2.52	0.27	18286	2.51	0.24	10957	2.5	0.23
1hx1B	31093	2.51	0.27	22868	2.5	0.26	14629	2.49	0.26
1y7qA	29251	2.53	0.27	19684	2.52	0.25	11106	2.51	0.24
1tubA	39286	2.53	0.22	35164	2.52	0.2	29722	2.52	0.19
5cbxA	29374	2.53	0.25	21693	2.52	0.23	14330	2.51	0.23
1nwqC	18753	2.46	0.34	7912	2.42	0.32	2925	2.35	0.33
> 0.4				> 0.5			> 0.6		
Protien	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}
2evaA	19036	2.53	0.2	11703	2.52	0.2	6146	2.52	0.21
3oe8C	15291	2.51	0.23	8152	2.49	0.23	3717	2.48	0.25
1oedB	8520	2.47	0.25	3782	2.46	0.26	1511	2.43	0.29
5l7gA	18108	2.52	0.22	10648	2.52	0.22	4996	2.51	0.23
1aifB	11690	2.53	0.2	5570	2.52	0.2	2345	2.52	0.2
1i4eB	17340	2.52	0.2	10078	2.52	0.2	5062	2.51	0.2
1oedA	7998	2.48	0.25	3432	2.45	0.26	1290	2.42	0.3
2ghuA	21474	2.51	0.19	14506	2.51	0.18	8791	2.51	0.18
5l7eA	18184	2.51	0.22	10616	2.51	0.22	5006	2.5	0.24
4n8eB	22695	2.52	0.2	13966	2.52	0.19	7393	2.51	0.2
1blxA	18298	2.53	0.21	10766	2.53	0.2	5303	2.52	0.21
3a20A	11439	2.5	0.21	5725	2.5	0.21	2555	2.49	0.21
3udwB	9207	2.52	0.2	4370	2.51	0.2	1840	2.51	0.21
4lgdG	887	2.28	0.38	301	2.17	0.41	118	2.1	0.44
3o47B	15595	2.52	0.21	8899	2.51	0.21	4375	2.51	0.22
2y83S	21871	2.52	0.19	14311	2.52	0.18	8100	2.51	0.19
2y83R	21411	2.52	0.19	13735	2.51	0.19	7668	2.51	0.19
3omvB	18524	2.53	0.2	11113	2.53	0.2	5573	2.52	0.21
5fd2A	17744	2.53	0.21	10663	2.52	0.21	5467	2.51	0.22

1yhwA	18755	2.51	0.21	11518	2.51	0.21	6233	2.51	0.21
1aboA	4804	2.49	0.2	2029	2.48	0.2	775	2.48	0.21
4qo1A	12690	2.51	0.19	7002	2.51	0.19	3425	2.51	0.19
4tq9B	15015	2.51	0.21	8699	2.51	0.2	4480	2.5	0.21
1s9iA	19732	2.53	0.2	12178	2.52	0.2	6456	2.52	0.21
1g50A	16701	2.52	0.22	9483	2.51	0.22	4531	2.5	0.24
4tntA	7950	2.5	0.23	3671	2.49	0.23	1493	2.48	0.25
1g50C	16845	2.52	0.22	9565	2.51	0.22	4528	2.5	0.23
2k7zA	14264	2.56	0.21	6831	2.56	0.2	2601	2.55	0.22
1ap2B	11902	2.51	0.2	6355	2.51	0.19	2887	2.5	0.19
1m2zA	18588	2.52	0.22	10913	2.51	0.22	5183	2.5	0.23
1a2vC	20256	2.53	0.2	13000	2.52	0.2	7069	2.52	0.2
1tupB	16371	2.52	0.2	9751	2.52	0.19	4928	2.52	0.2
1jaiA	15397	2.52	0.2	9037	2.51	0.2	4657	2.51	0.2
1bmfE	21123	2.52	0.2	13473	2.51	0.2	7531	2.51	0.2
6q21B	15979	2.51	0.2	9576	2.51	0.2	5063	2.5	0.2
4obeA	14849	2.51	0.21	8540	2.5	0.21	4394	2.5	0.21
1dykA	19971	2.51	0.19	12243	2.5	0.18	6567	2.5	0.18
1qupA	14644	2.51	0.21	8141	2.51	0.2	3964	2.51	0.2
1nobF	15467	2.51	0.19	8607	2.51	0.19	4137	2.5	0.19
4dsoA	15058	2.51	0.21	8734	2.51	0.21	4571	2.5	0.22
4eklA	19667	2.53	0.2	12014	2.52	0.2	6224	2.52	0.2
5hheA	1086	2.18	0.49	367	2.03	0.56	154	1.94	0.57
1nvwR	14928	2.52	0.21	8648	2.51	0.21	4294	2.51	0.22
4csjA	17271	2.53	0.21	9932	2.52	0.21	4757	2.51	0.22
1g5mA	14821	2.52	0.21	7829	2.52	0.21	3516	2.51	0.22
3ldqB	7488	2.48	0.27	3279	2.45	0.29	1335	2.42	0.33
2pqfA	15378	2.51	0.2	8833	2.51	0.2	4436	2.5	0.21
4lqyA	21485	2.52	0.19	14039	2.52	0.19	8066	2.51	0.19

1he8B	15549	2.51	0.21	9156	2.5	0.2	4839	2.5	0.21
4p6xC	17888	2.52	0.22	10440	2.52	0.22	5021	2.51	0.22
5l7hA	18054	2.52	0.22	10480	2.51	0.22	4924	2.5	0.24
1yjdC	11693	2.51	0.2	6066	2.51	0.2	2813	2.5	0.2
3jckG	8175	2.5	0.24	3960	2.49	0.24	1788	2.49	0.24
1xd2C	17918	2.52	0.23	10181	2.51	0.23	4924	2.49	0.24
4udcA	17965	2.53	0.21	10356	2.52	0.22	4947	2.51	0.22
3szaB	19560	2.52	0.21	12605	2.51	0.21	7100	2.51	0.21
4p6xA	18212	2.52	0.22	10717	2.52	0.22	5248	2.51	0.23
1hurA	15879	2.52	0.2	9000	2.51	0.2	4405	2.51	0.2
4jupB	7966	2.51	0.23	3577	2.5	0.23	1413	2.49	0.23
1na7A	20930	2.53	0.2	13018	2.53	0.2	6828	2.52	0.21
7catB	17637	2.52	0.2	10075	2.52	0.2	4737	2.51	0.2
1qduA	11271	2.5	0.22	5617	2.49	0.22	2437	2.48	0.23
3mupA	8402	2.51	0.22	3932	2.51	0.22	1744	2.5	0.22
2m1wA	12042	2.52	0.21	6260	2.51	0.21	2843	2.5	0.22
4j93A	12048	2.49	0.23	6278	2.48	0.24	2911	2.47	0.26
4pf3A	17770	2.51	0.23	10216	2.5	0.23	4741	2.49	0.25
3zzxB	10565	2.5	0.23	5416	2.49	0.23	2441	2.48	0.24
2kecA	4057	2.48	0.24	1453	2.47	0.24	518	2.44	0.27
6q21D	15780	2.51	0.2	9271	2.51	0.2	4758	2.51	0.2
2aa2A	17521	2.51	0.23	10111	2.51	0.23	4751	2.49	0.25
2jcwA	13693	2.49	0.19	7954	2.49	0.19	4081	2.49	0.19
1r6qA	12775	2.5	0.23	6913	2.49	0.23	3383	2.48	0.24
2geqA	12631	2.52	0.21	6904	2.52	0.21	3329	2.51	0.21
1e96A	15128	2.52	0.21	8855	2.52	0.2	4685	2.51	0.2
5g3jA	17552	2.53	0.21	9973	2.52	0.22	4651	2.51	0.23
4jnkA	19403	2.51	0.21	12216	2.5	0.21	6766	2.49	0.21
1b34A	5675	2.49	0.22	2407	2.48	0.22	958	2.47	0.23

1hx1B	7884	2.47	0.27	3466	2.44	0.29	1370	2.4	0.32
1y7qA	5065	2.49	0.24	1809	2.47	0.26	625	2.44	0.28
1tubA	23154	2.52	0.19	15900	2.51	0.18	9471	2.51	0.19
5cbxA	8132	2.5	0.23	3843	2.49	0.23	1635	2.48	0.24
1nwqC	979	2.27	0.36	339	2.17	0.41	166	2.1	0.43
> 0.7				> 0.8			> 0.9		
Protien	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}
2evaA	2165	2.51	0.23	441	2.47	0.27	70	2.35	0.38
3oe8C	1261	2.45	0.29	264	2.38	0.34	44	2.25	0.36
1oedB	443	2.38	0.34	101	2.24	0.46	17	1.85	0.67
5l7gA	1536	2.48	0.26	259	2.44	0.36	43	2.28	0.54
1aifB	767	2.51	0.21	157	2.52	0.25	32	2.5	0.29
1i4eB	1742	2.51	0.22	364	2.49	0.27	51	2.39	0.48
1oedA	384	2.34	0.37	83	2.13	0.53	22	1.94	0.72
2ghuA	4002	2.51	0.19	1112	2.51	0.2	161	2.48	0.24
5l7eA	1588	2.47	0.28	291	2.4	0.37	33	2.17	0.54
4n8eB	2836	2.51	0.21	696	2.49	0.25	127	2.44	0.35
1blxA	1743	2.51	0.24	343	2.49	0.31	44	2.39	0.37
3a20A	859	2.48	0.22	164	2.43	0.28	30	2.36	0.32
3udwB	546	2.49	0.22	100	2.46	0.26	15	2.28	0.39
4lgdG	42	1.93	0.48	18	1.79	0.57	8	1.64	0.73
3o47B	1599	2.5	0.24	365	2.48	0.29	60	2.4	0.42
2y83S	3408	2.51	0.2	823	2.5	0.23	140	2.45	0.32
2y83R	3134	2.5	0.2	796	2.49	0.24	118	2.47	0.35
3omvB	1876	2.51	0.23	359	2.46	0.27	47	2.36	0.41
5fd2A	2007	2.51	0.25	408	2.47	0.35	58	2.23	0.57
1yhwa	2469	2.5	0.23	537	2.47	0.28	71	2.41	0.4
1aboA	235	2.47	0.2	40	2.43	0.21	8	2.46	0.26
4qo1A	1212	2.51	0.2	267	2.53	0.21	38	2.51	0.29

4tq9B	1704	2.5	0.22	373	2.47	0.27	59	2.37	0.35
1s9iA	2329	2.51	0.23	509	2.48	0.29	65	2.3	0.53
1g50A	1427	2.46	0.3	278	2.37	0.42	49	2.1	0.63
4tntA	466	2.46	0.28	99	2.36	0.38	19	2.25	0.52
1g50C	1427	2.48	0.27	247	2.42	0.38	37	2.21	0.5
2k7zA	620	2.52	0.25	105	2.49	0.31	16	2.36	0.41
1ap2B	1004	2.5	0.21	216	2.49	0.29	31	2.39	0.52
1m2zA	1703	2.48	0.26	319	2.39	0.37	36	2.18	0.59
1a2vC	2703	2.51	0.22	576	2.5	0.27	81	2.43	0.42
1tupB	1726	2.51	0.22	339	2.49	0.3	40	2.34	0.58
1jaiA	1768	2.51	0.21	414	2.52	0.23	72	2.51	0.28
1bmfE	2926	2.5	0.21	687	2.48	0.26	120	2.44	0.36
6q21B	1934	2.5	0.21	429	2.47	0.24	55	2.43	0.31
4obeA	1644	2.49	0.23	386	2.48	0.3	62	2.35	0.43
1dykA	2488	2.49	0.19	542	2.49	0.2	79	2.47	0.21
1qupA	1448	2.49	0.21	298	2.47	0.22	42	2.45	0.22
1nobF	1370	2.49	0.19	280	2.48	0.21	41	2.43	0.24
4dsoA	1743	2.49	0.24	423	2.46	0.31	56	2.33	0.4
4eklA	2274	2.51	0.22	401	2.47	0.28	55	2.35	0.43
5hheA	75	1.82	0.61	34	1.57	0.59	17	1.32	0.55
1nvwR	1607	2.5	0.22	341	2.5	0.29	45	2.37	0.46
4csjA	1465	2.48	0.25	252	2.43	0.34	35	2.33	0.47
1g5mA	1020	2.5	0.25	195	2.46	0.32	26	2.3	0.43
3ldqB	409	2.32	0.42	108	2.15	0.58	31	1.85	0.75
2pqfA	1592	2.49	0.24	351	2.46	0.31	53	2.28	0.61
4lqyA	3199	2.51	0.2	731	2.49	0.22	124	2.46	0.27
1he8B	1882	2.5	0.21	431	2.47	0.25	65	2.41	0.28
4p6xC	1512	2.48	0.27	271	2.4	0.32	41	2.22	0.34
5l7hA	1484	2.47	0.28	248	2.39	0.36	38	2.16	0.47

1yjdC	935	2.49	0.23	192	2.46	0.3	30	2.47	0.55
3jckG	626	2.46	0.27	159	2.43	0.34	24	2.31	0.43
1xd2C	1664	2.46	0.29	356	2.37	0.38	53	2.2	0.59
4udcA	1535	2.5	0.25	270	2.44	0.34	39	2.37	0.41
3szaB	2931	2.5	0.23	719	2.49	0.27	101	2.44	0.41
4p6xA	1670	2.49	0.26	308	2.4	0.35	39	2.2	0.51
1hurA	1503	2.5	0.21	305	2.48	0.24	29	2.37	0.25
4jupB	380	2.48	0.26	59	2.41	0.31	8	2.4	0.22
1na7A	2439	2.51	0.23	479	2.45	0.3	59	2.39	0.34
7catB	1513	2.5	0.21	281	2.47	0.26	49	2.44	0.32
1qduA	729	2.47	0.26	164	2.42	0.35	32	2.19	0.53
3mupA	587	2.5	0.24	130	2.47	0.28	16	2.46	0.37
2m1wA	947	2.48	0.25	191	2.43	0.32	35	2.26	0.35
4j93A	964	2.44	0.3	218	2.36	0.39	43	2.32	0.52
4pf3A	1470	2.46	0.3	272	2.34	0.42	42	2.13	0.57
3zzxB	821	2.45	0.27	164	2.41	0.35	22	2.36	0.5
2kecA	153	2.45	0.27	28	2.35	0.35	4	2.18	0.35
6q21D	1826	2.51	0.21	404	2.49	0.25	55	2.41	0.34
2aa2A	1421	2.45	0.31	252	2.35	0.39	33	2.14	0.56
2jcwA	1572	2.48	0.19	369	2.48	0.21	37	2.44	0.23
1r6qA	1178	2.47	0.29	280	2.41	0.41	43	2.07	0.72
2geqA	1170	2.5	0.23	236	2.46	0.3	34	2.34	0.43
1e96A	1779	2.51	0.23	432	2.49	0.27	67	2.36	0.42
5g3jA	1425	2.49	0.27	246	2.41	0.36	39	2.11	0.58
4jnkA	2818	2.48	0.23	693	2.45	0.3	92	2.39	0.42
1b34A	278	2.45	0.25	56	2.44	0.24	12	2.38	0.24
1hx1B	479	2.34	0.39	118	2.1	0.49	27	1.81	0.53
1y7qA	163	2.38	0.34	31	2.19	0.53	6	1.65	0.79
1tubA	3952	2.51	0.19	896	2.5	0.23	115	2.43	0.3

5cbxA	546	2.47	0.26	108	2.44	0.3	18	2.35	0.34
1nwqC	66	1.97	0.45	31	1.89	0.49	14	1.66	0.45

Table A.2: Filtering interface structures according to percentage contact match.

#int: Number of interface structures which passes the filter M_{RMSD} : Average RMSD σ_{RMSD} : standard deviation for RMSD

Protien	#int	> 0.1		#int	> 0.2		#int	> 0.3	
		M_{RMSD}	σ_{RMSD}		M_{RMSD}	σ_{RMSD}		M_{RMSD}	σ_{RMSD}
2evaA	18632	2.53	0.23	4877	2.53	0.24	1237	2.52	0.26
3oe8C	20885	2.52	0.25	6458	2.5	0.25	1853	2.49	0.27
1oedB	20130	2.5	0.27	7697	2.49	0.27	2556	2.47	0.28
5l7gA	16831	2.52	0.24	4320	2.51	0.26	1128	2.49	0.27
1aifB	21345	2.54	0.23	7397	2.53	0.22	2167	2.53	0.23
1i4eB	17967	2.53	0.23	4909	2.52	0.23	1246	2.51	0.24
1oedA	20056	2.51	0.27	7642	2.49	0.27	2537	2.47	0.28
2ghuA	16017	2.53	0.22	3781	2.53	0.23	929	2.53	0.23
5l7eA	17068	2.52	0.24	4361	2.5	0.26	1075	2.49	0.27
4n8eB	19189	2.53	0.22	4842	2.52	0.23	1179	2.52	0.25
1blxA	19677	2.54	0.23	5327	2.53	0.23	1400	2.53	0.24
3a20A	17551	2.52	0.24	5176	2.51	0.24	1413	2.5	0.24
3udwB	15882	2.52	0.22	4713	2.51	0.22	1246	2.51	0.22
4lgdG	10223	2.46	0.35	3208	2.41	0.35	1024	2.39	0.36
3o47B	18961	2.53	0.24	5343	2.51	0.25	1462	2.51	0.24
2y83S	18281	2.53	0.22	4627	2.53	0.23	1161	2.53	0.24
2y83R	17248	2.53	0.22	4162	2.52	0.23	1036	2.52	0.25
3omvB	19852	2.54	0.23	5412	2.54	0.23	1400	2.54	0.24
5fd2A	19101	2.54	0.23	4998	2.53	0.25	1283	2.52	0.27

1yhwA	19169	2.52	0.24	5053	2.51	0.25	1268	2.5	0.25
1aboA	11285	2.51	0.24	3352	2.5	0.23	882	2.49	0.22
4qo1A	17780	2.52	0.22	5313	2.52	0.22	1407	2.51	0.22
4tq9B	17167	2.52	0.24	4614	2.5	0.26	1124	2.49	0.27
1s9iA	18835	2.54	0.23	4938	2.53	0.24	1273	2.53	0.24
1g50A	17742	2.52	0.24	4764	2.51	0.26	1204	2.49	0.27
4tntA	16774	2.52	0.25	5616	2.5	0.25	1662	2.5	0.25
1g50C	17419	2.52	0.24	4701	2.51	0.26	1183	2.49	0.27
2k7zA	21802	2.57	0.23	6972	2.56	0.23	1960	2.56	0.23
1ap2B	17109	2.52	0.22	4994	2.52	0.21	1384	2.51	0.23
1m2zA	16922	2.52	0.24	4225	2.52	0.26	1095	2.51	0.28
1a2vC	17816	2.53	0.24	4776	2.51	0.25	1222	2.51	0.26
1tupB	17760	2.53	0.23	4796	2.52	0.23	1266	2.52	0.25
1jaiA	15467	2.52	0.24	3744	2.51	0.25	921	2.5	0.25
1bmfE	19026	2.52	0.23	5187	2.52	0.24	1370	2.52	0.25
6q21B	18439	2.52	0.23	5085	2.51	0.24	1365	2.5	0.25
4obeA	16383	2.52	0.25	4222	2.5	0.26	1063	2.49	0.27
1dykA	20019	2.52	0.21	5510	2.51	0.22	1464	2.5	0.23
1qupA	19548	2.53	0.23	5765	2.52	0.23	1597	2.52	0.23
1nobF	19902	2.52	0.22	6004	2.52	0.22	1645	2.52	0.22
4dsoA	17344	2.52	0.25	4729	2.5	0.26	1304	2.48	0.29
4eklA	18394	2.54	0.23	4749	2.53	0.24	1228	2.52	0.25
5hheA	10480	2.43	0.39	3413	2.36	0.41	1174	2.32	0.43
1nvwR	17426	2.53	0.24	4703	2.52	0.25	1184	2.51	0.25
4csjA	17788	2.53	0.24	4882	2.52	0.25	1280	2.51	0.27
1g5mA	18741	2.53	0.23	5165	2.52	0.24	1264	2.51	0.25
3ldqB	15482	2.5	0.28	4633	2.47	0.3	1265	2.46	0.33
2pqfA	19184	2.53	0.23	5611	2.52	0.24	1508	2.51	0.26
4lqyA	16996	2.52	0.23	4410	2.51	0.24	1149	2.5	0.26

1he8B	16201	2.52	0.24	3976	2.5	0.25	978	2.49	0.27
4p6xC	17340	2.53	0.24	4430	2.52	0.25	1129	2.51	0.26
5l7hA	17013	2.52	0.24	4249	2.5	0.25	1058	2.5	0.26
1yjdC	17476	2.52	0.22	5272	2.51	0.22	1433	2.51	0.24
3jckG	18566	2.52	0.28	6208	2.5	0.28	2032	2.47	0.29
1xd2C	19937	2.53	0.25	5357	2.51	0.26	1436	2.49	0.29
4udcA	17466	2.53	0.24	4568	2.51	0.25	1170	2.51	0.27
3szaB	17267	2.52	0.24	4611	2.5	0.26	1216	2.49	0.29
4p6xA	17679	2.53	0.24	4435	2.52	0.25	1111	2.51	0.27
1hurA	16862	2.53	0.24	4218	2.52	0.24	1042	2.52	0.25
4jupB	13944	2.52	0.26	3715	2.51	0.25	972	2.49	0.26
1na7A	17681	2.54	0.22	4193	2.53	0.24	1055	2.51	0.25
7catB	20137	2.53	0.24	5721	2.52	0.24	1550	2.52	0.25
1qduA	16960	2.51	0.25	4804	2.5	0.25	1255	2.49	0.26
3mupA	15726	2.52	0.26	4819	2.51	0.26	1359	2.49	0.26
2m1wA	17581	2.52	0.25	5264	2.5	0.26	1473	2.48	0.27
4j93A	17206	2.51	0.26	4793	2.49	0.27	1260	2.47	0.28
4pf3A	18352	2.52	0.25	4850	2.5	0.27	1364	2.49	0.3
3zzxB	14887	2.51	0.26	3912	2.49	0.27	981	2.49	0.3
2kecA	13614	2.51	0.29	4170	2.49	0.28	1213	2.47	0.27
6q21D	18181	2.52	0.24	4963	2.51	0.25	1361	2.49	0.26
2aa2A	18899	2.52	0.25	5199	2.5	0.27	1356	2.47	0.27
2jcwA	18200	2.5	0.22	5463	2.5	0.22	1476	2.49	0.23
1r6qA	17664	2.5	0.26	5103	2.48	0.28	1388	2.46	0.31
2geqA	18175	2.53	0.25	5279	2.51	0.26	1408	2.49	0.28
1e96A	19258	2.53	0.24	5873	2.52	0.24	1597	2.51	0.25
5g3jA	17619	2.53	0.25	4630	2.52	0.26	1163	2.5	0.28
4jnkA	18940	2.52	0.24	5130	2.51	0.26	1361	2.5	0.28
1b34A	14187	2.51	0.26	4286	2.5	0.26	1167	2.49	0.25

1hx1B	16561	2.5	0.28	5212	2.47	0.29	1498	2.44	0.3
1y7qA	17463	2.52	0.27	6075	2.5	0.26	1886	2.5	0.27
1tubA	18566	2.53	0.22	5115	2.52	0.23	1434	2.51	0.25
5cbxA	16143	2.52	0.25	5131	2.51	0.25	1456	2.49	0.25
1nwqC	10592	2.45	0.35	3476	2.41	0.35	1113	2.36	0.37
> 0.4				> 0.5			> 0.6		
Protien	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}
2evaA	292	2.49	0.3	58	2.45	0.34	9	2.4	0.39
3oe8C	485	2.47	0.27	83	2.43	0.27	25	2.41	0.27
1oedB	714	2.46	0.29	143	2.48	0.32	51	2.43	0.36
5l7gA	272	2.49	0.29	49	2.5	0.27	15	2.54	0.3
1aifB	579	2.52	0.22	105	2.49	0.24	35	2.47	0.31
1i4eB	299	2.5	0.26	59	2.45	0.4	22	2.44	0.25
1oedA	762	2.44	0.31	172	2.38	0.37	48	2.41	0.35
2ghuA	223	2.49	0.25	29	2.47	0.18	8	2.44	0.17
5l7eA	249	2.48	0.27	36	2.46	0.27	9	2.41	0.18
4n8eB	278	2.54	0.25	55	2.5	0.3	15	2.46	0.27
1blxA	351	2.52	0.23	66	2.44	0.22	17	2.42	0.22
3a20A	373	2.49	0.25	58	2.44	0.29	20	2.44	0.32
3udwB	317	2.5	0.22	56	2.47	0.23	18	2.5	0.13
4lgdG	284	2.33	0.39	60	2.29	0.4	22	2.24	0.43
3o47B	341	2.5	0.25	59	2.5	0.22	13	2.45	0.29
2y83S	292	2.53	0.28	46	2.54	0.28	6	2.6	0.22
2y83R	231	2.48	0.28	32	2.51	0.16	8	2.54	0.06
3omvB	324	2.52	0.25	53	2.57	0.28	14	2.51	0.39
5fd2A	319	2.5	0.3	52	2.44	0.41	13	2.41	0.4
1yhwa	286	2.5	0.26	46	2.5	0.29	15	2.56	0.31
1aboA	219	2.49	0.21	36	2.47	0.2	8	2.54	0.22
4qo1A	327	2.51	0.24	58	2.48	0.29	11	2.46	0.4

4tq9B	289	2.49	0.26	52	2.5	0.31	22	2.54	0.3
1s9iA	281	2.52	0.28	54	2.47	0.44	21	2.6	0.29
1g50A	254	2.46	0.31	56	2.38	0.41	20	2.46	0.45
4tntA	438	2.48	0.25	90	2.45	0.28	27	2.43	0.35
1g50C	291	2.49	0.28	41	2.45	0.25	13	2.35	0.33
2k7zA	501	2.56	0.23	98	2.52	0.26	31	2.48	0.26
1ap2B	359	2.51	0.26	57	2.46	0.39	16	2.44	0.22
1m2zA	263	2.5	0.29	43	2.47	0.32	13	2.55	0.22
1a2vC	298	2.5	0.26	42	2.47	0.32	12	2.35	0.5
1tupB	314	2.52	0.3	67	2.55	0.37	21	2.5	0.58
1jaiA	207	2.5	0.28	37	2.51	0.37	7	2.61	0.15
1bmfE	308	2.49	0.27	47	2.51	0.27	19	2.5	0.3
6q21B	340	2.48	0.28	59	2.43	0.25	18	2.38	0.25
4obeA	259	2.49	0.27	44	2.45	0.23	16	2.51	0.2
1dykA	358	2.49	0.25	46	2.38	0.36	15	2.3	0.29
1qupA	372	2.51	0.25	61	2.48	0.26	17	2.47	0.22
1nobF	390	2.51	0.22	82	2.52	0.23	31	2.52	0.19
4dsoA	345	2.47	0.29	57	2.45	0.32	17	2.44	0.29
4eklA	260	2.5	0.29	48	2.49	0.22	12	2.53	0.22
5hheA	351	2.25	0.48	82	2.16	0.49	27	2.07	0.53
1nvwR	272	2.5	0.25	52	2.48	0.23	18	2.47	0.26
4csjA	317	2.49	0.27	49	2.53	0.26	12	2.56	0.2
1g5mA	317	2.5	0.26	53	2.48	0.27	19	2.53	0.18
3ldqB	317	2.42	0.36	54	2.33	0.5	13	2.12	0.45
2pqfA	354	2.5	0.32	68	2.43	0.38	21	2.31	0.59
4lqyA	259	2.49	0.27	55	2.43	0.32	16	2.37	0.41
1he8B	227	2.49	0.27	39	2.44	0.28	7	2.47	0.19
4p6xC	241	2.5	0.27	35	2.49	0.3	5	2.62	0.15
5l7hA	262	2.48	0.29	43	2.4	0.29	15	2.37	0.26

1yjdC	340	2.5	0.26	76	2.51	0.24	23	2.52	0.26
3jckG	560	2.44	0.3	117	2.42	0.29	41	2.38	0.31
1xd2C	344	2.43	0.34	59	2.37	0.34	20	2.41	0.28
4udcA	281	2.48	0.28	56	2.48	0.27	19	2.42	0.35
3szaB	307	2.47	0.33	45	2.41	0.41	17	2.46	0.36
4p6xA	267	2.48	0.3	42	2.5	0.31	13	2.52	0.35
1hurA	242	2.53	0.25	33	2.6	0.27	12	2.54	0.31
4jupB	240	2.5	0.3	35	2.49	0.28	9	2.47	0.33
1na7A	258	2.48	0.25	37	2.52	0.26	12	2.63	0.17
7catB	418	2.51	0.26	69	2.45	0.27	21	2.42	0.24
1qduA	287	2.48	0.27	61	2.44	0.38	18	2.4	0.62
3mupA	358	2.49	0.27	57	2.47	0.27	19	2.49	0.31
2m1wA	378	2.46	0.27	65	2.43	0.28	19	2.47	0.22
4j93A	314	2.46	0.28	63	2.43	0.34	17	2.41	0.29
4pf3A	323	2.47	0.31	55	2.41	0.39	17	2.36	0.29
3zzxB	227	2.48	0.3	42	2.51	0.28	3	2.5	0.16
2kecA	273	2.44	0.28	44	2.42	0.34	13	2.29	0.42
6q21D	320	2.47	0.3	53	2.48	0.31	14	2.48	0.23
2aa2A	326	2.47	0.3	58	2.49	0.29	11	2.4	0.42
2jcwA	383	2.49	0.23	68	2.54	0.22	21	2.61	0.21
1r6qA	354	2.41	0.38	65	2.4	0.39	21	2.38	0.36
2geqA	366	2.49	0.29	62	2.51	0.31	20	2.42	0.28
1e96A	391	2.51	0.29	78	2.5	0.33	22	2.39	0.5
5g3jA	303	2.46	0.29	52	2.4	0.43	15	2.28	0.54
4jnkA	320	2.49	0.29	66	2.51	0.3	26	2.44	0.32
1b34A	292	2.48	0.25	55	2.48	0.24	9	2.51	0.23
1hx1B	386	2.44	0.33	89	2.38	0.37	34	2.4	0.3
1y7qA	538	2.48	0.28	132	2.47	0.27	47	2.46	0.3
1tubA	342	2.51	0.26	58	2.49	0.27	12	2.43	0.45

5cbxA	369	2.48	0.23	58	2.43	0.26	19	2.43	0.25
1nwqC	341	2.33	0.41	73	2.24	0.37	26	2.18	0.44
> 0.4				> 0.5			> 0.6		
Protien	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}
2evaA	292	2.49	0.3	58	2.45	0.34	9	2.4	0.39
3oe8C	485	2.47	0.27	83	2.43	0.27	25	2.41	0.27
1oedB	714	2.46	0.29	143	2.48	0.32	51	2.43	0.36
5l7gA	272	2.49	0.29	49	2.5	0.27	15	2.54	0.3
1aifB	579	2.52	0.22	105	2.49	0.24	35	2.47	0.31
1i4eB	299	2.5	0.26	59	2.45	0.4	22	2.44	0.25
1oedA	762	2.44	0.31	172	2.38	0.37	48	2.41	0.35
2ghuA	223	2.49	0.25	29	2.47	0.18	8	2.44	0.17
5l7eA	249	2.48	0.27	36	2.46	0.27	9	2.41	0.18
4n8eB	278	2.54	0.25	55	2.5	0.3	15	2.46	0.27
1blxA	351	2.52	0.23	66	2.44	0.22	17	2.42	0.22
3a20A	373	2.49	0.25	58	2.44	0.29	20	2.44	0.32
3udwB	317	2.5	0.22	56	2.47	0.23	18	2.5	0.13
4lgdG	284	2.33	0.39	60	2.29	0.4	22	2.24	0.43
3o47B	341	2.5	0.25	59	2.5	0.22	13	2.45	0.29
2y83S	292	2.53	0.28	46	2.54	0.28	6	2.6	0.22
2y83R	231	2.48	0.28	32	2.51	0.16	8	2.54	0.06
3omvB	324	2.52	0.25	53	2.57	0.28	14	2.51	0.39
5fd2A	319	2.5	0.3	52	2.44	0.41	13	2.41	0.4
1yhwa	286	2.5	0.26	46	2.5	0.29	15	2.56	0.31
1aboA	219	2.49	0.21	36	2.47	0.2	8	2.54	0.22
4qo1A	327	2.51	0.24	58	2.48	0.29	11	2.46	0.4
4tq9B	289	2.49	0.26	52	2.5	0.31	22	2.54	0.3
1s9iA	281	2.52	0.28	54	2.47	0.44	21	2.6	0.29
1g50A	254	2.46	0.31	56	2.38	0.41	20	2.46	0.45

4tntA	438	2.48	0.25	90	2.45	0.28	27	2.43	0.35
1g50C	291	2.49	0.28	41	2.45	0.25	13	2.35	0.33
2k7zA	501	2.56	0.23	98	2.52	0.26	31	2.48	0.26
1ap2B	359	2.51	0.26	57	2.46	0.39	16	2.44	0.22
1m2zA	263	2.5	0.29	43	2.47	0.32	13	2.55	0.22
1a2vC	298	2.5	0.26	42	2.47	0.32	12	2.35	0.5
1tupB	314	2.52	0.3	67	2.55	0.37	21	2.5	0.58
1jaiA	207	2.5	0.28	37	2.51	0.37	7	2.61	0.15
1bmfE	308	2.49	0.27	47	2.51	0.27	19	2.5	0.3
6q21B	340	2.48	0.28	59	2.43	0.25	18	2.38	0.25
4obeA	259	2.49	0.27	44	2.45	0.23	16	2.51	0.2
1dykA	358	2.49	0.25	46	2.38	0.36	15	2.3	0.29
1qupA	372	2.51	0.25	61	2.48	0.26	17	2.47	0.22
1nobF	390	2.51	0.22	82	2.52	0.23	31	2.52	0.19
4dsoA	345	2.47	0.29	57	2.45	0.32	17	2.44	0.29
4eklA	260	2.5	0.29	48	2.49	0.22	12	2.53	0.22
5hheA	351	2.25	0.48	82	2.16	0.49	27	2.07	0.53
1nvwR	272	2.5	0.25	52	2.48	0.23	18	2.47	0.26
4csjA	317	2.49	0.27	49	2.53	0.26	12	2.56	0.2
1g5mA	317	2.5	0.26	53	2.48	0.27	19	2.53	0.18
3ldqB	317	2.42	0.36	54	2.33	0.5	13	2.12	0.45
2pqfA	354	2.5	0.32	68	2.43	0.38	21	2.31	0.59
4lqyA	259	2.49	0.27	55	2.43	0.32	16	2.37	0.41
1he8B	227	2.49	0.27	39	2.44	0.28	7	2.47	0.19
4p6xC	241	2.5	0.27	35	2.49	0.3	5	2.62	0.15
5l7hA	262	2.48	0.29	43	2.4	0.29	15	2.37	0.26
1yjdC	340	2.5	0.26	76	2.51	0.24	23	2.52	0.26
3jckG	560	2.44	0.3	117	2.42	0.29	41	2.38	0.31
1xd2C	344	2.43	0.34	59	2.37	0.34	20	2.41	0.28

4udcA	281	2.48	0.28	56	2.48	0.27	19	2.42	0.35
3szaB	307	2.47	0.33	45	2.41	0.41	17	2.46	0.36
4p6xA	267	2.48	0.3	42	2.5	0.31	13	2.52	0.35
1hurA	242	2.53	0.25	33	2.6	0.27	12	2.54	0.31
4jupB	240	2.5	0.3	35	2.49	0.28	9	2.47	0.33
1na7A	258	2.48	0.25	37	2.52	0.26	12	2.63	0.17
7catB	418	2.51	0.26	69	2.45	0.27	21	2.42	0.24
1qduA	287	2.48	0.27	61	2.44	0.38	18	2.4	0.62
3mupA	358	2.49	0.27	57	2.47	0.27	19	2.49	0.31
2m1wA	378	2.46	0.27	65	2.43	0.28	19	2.47	0.22
4j93A	314	2.46	0.28	63	2.43	0.34	17	2.41	0.29
4pf3A	323	2.47	0.31	55	2.41	0.39	17	2.36	0.29
3zzxB	227	2.48	0.3	42	2.51	0.28	3	2.5	0.16
2kecA	273	2.44	0.28	44	2.42	0.34	13	2.29	0.42
6q21D	320	2.47	0.3	53	2.48	0.31	14	2.48	0.23
2aa2A	326	2.47	0.3	58	2.49	0.29	11	2.4	0.42
2jcwA	383	2.49	0.23	68	2.54	0.22	21	2.61	0.21
1r6qA	354	2.41	0.38	65	2.4	0.39	21	2.38	0.36
2geqA	366	2.49	0.29	62	2.51	0.31	20	2.42	0.28
1e96A	391	2.51	0.29	78	2.5	0.33	22	2.39	0.5
5g3jA	303	2.46	0.29	52	2.4	0.43	15	2.28	0.54
4jnkA	320	2.49	0.29	66	2.51	0.3	26	2.44	0.32
1b34A	292	2.48	0.25	55	2.48	0.24	9	2.51	0.23
1hx1B	386	2.44	0.33	89	2.38	0.37	34	2.4	0.3
1y7qA	538	2.48	0.28	132	2.47	0.27	47	2.46	0.3
1tubA	342	2.51	0.26	58	2.49	0.27	12	2.43	0.45
5cbxA	369	2.48	0.23	58	2.43	0.26	19	2.43	0.25
1nwqC	341	2.33	0.41	73	2.24	0.37	26	2.18	0.44
> 0.7				> 0.8				> 0.9	

Protien	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}	#int	M_{RMSD}	σ_{RMSD}
2evaA	2	2.44	0.16	1	2.6	0	0	nan	nan
3oe8C	8	2.42	0.32	1	1.89	0	0	nan	nan
1oedB	15	2.35	0.53	7	2.07	0.63	1	1.02	0
5l7gA	7	2.44	0.3	1	2.54	0	0	nan	nan
1aifB	9	2.44	0.43	4	2.6	0.3	1	3.08	0
1i4eB	7	2.46	0.28	1	2.11	0	0	nan	nan
1oedA	5	2.25	0.44	1	2.34	0	1	2.34	0
2ghuA	3	2.31	0.13	0	nan	nan	0	nan	nan
5l7eA	4	2.48	0.13	0	nan	nan	0	nan	nan
4n8eB	3	2.45	0.09	2	2.51	0	0	nan	nan
1blxA	7	2.32	0.16	3	2.32	0.22	0	nan	nan
3a20A	6	2.51	0.24	1	2.43	0	0	nan	nan
3udwB	3	2.56	0.11	0	nan	nan	0	nan	nan
4lgdG	5	2.36	0.48	1	2.18	0	0	nan	nan
3o47B	5	2.55	0.28	1	2.26	0	0	nan	nan
2y83S	0	nan	nan	0	nan	nan	0	nan	nan
2y83R	1	2.47	0	0	nan	nan	0	nan	nan
3omvB	6	2.54	0.57	2	1.94	0.61	0	nan	nan
5fd2A	2	2.53	0.01	0	nan	nan	0	nan	nan
1yhwa	3	2.66	0.25	1	2.57	0	0	nan	nan
1aboA	3	2.6	0.26	1	2.28	0	0	nan	nan
4qo1A	1	2.67	0	0	nan	nan	0	nan	nan
4tq9B	7	2.52	0.21	0	nan	nan	0	nan	nan
1s9iA	6	2.68	0.37	2	2.85	0.09	0	nan	nan
1g50A	6	2.47	0.36	0	nan	nan	0	nan	nan
4tntA	4	2.26	0.58	0	nan	nan	0	nan	nan
1g50C	6	2.23	0.34	0	nan	nan	0	nan	nan

2k7zA	12	2.51	0.25	4	2.44	0.09	0	nan	nan
1ap2B	7	2.33	0.22	0	nan	nan	0	nan	nan
1m2zA	2	2.53	0.09	1	2.44	0	0	nan	nan
1a2vC	5	2.25	0.74	0	nan	nan	0	nan	nan
1tupB	10	2.46	0.82	1	2.66	0	0	nan	nan
1jaiA	4	2.63	0.18	2	2.77	0.15	1	2.93	0
1bmfE	6	2.5	0.2	2	2.46	0.14	0	nan	nan
6q21B	7	2.34	0.26	3	2.15	0.2	0	nan	nan
4obeA	6	2.56	0.24	2	2.74	0.24	0	nan	nan
1dykA	3	2.35	0.12	0	nan	nan	0	nan	nan
1qupA	7	2.54	0.16	1	2.47	0	0	nan	nan
1nobF	6	2.49	0.16	0	nan	nan	0	nan	nan
4dsoA	4	2.42	0.28	0	nan	nan	0	nan	nan
4eklA	2	2.42	0.1	1	2.52	0	1	2.52	0
5hheA	13	2.08	0.43	1	1.61	0	1	1.61	0
1nvwR	3	2.63	0.19	1	2.84	0	0	nan	nan
4csjA	2	2.75	0.21	0	nan	nan	0	nan	nan
1g5mA	5	2.49	0.1	3	2.47	0.1	2	2.54	0.04
3ldqB	6	2.21	0.46	2	1.73	0.43	1	2.16	0
2pqfA	10	2.09	0.78	7	2.28	0.4	2	1.76	0.36
4lqyA	6	2.54	0.18	0	nan	nan	0	nan	nan
1he8B	1	2.63	0	1	2.63	0	0	nan	nan
4p6xC	1	2.88	0	0	nan	nan	0	nan	nan
5l7hA	3	2.42	0.06	0	nan	nan	0	nan	nan
1yjdC	9	2.51	0.16	3	2.42	0.22	0	nan	nan
3jckG	12	2.37	0.28	2	2.52	0.4	0	nan	nan
1xd2C	5	2.4	0.16	0	nan	nan	0	nan	nan
4udcA	9	2.37	0.42	2	2.52	0.16	0	nan	nan
3szaB	3	2.49	0.39	0	nan	nan	0	nan	nan

4p6xA	2	2.11	0.03	0	nan	nan	0	nan	nan
1hurA	6	2.44	0.3	0	nan	nan	0	nan	nan
4jupB	4	2.5	0.2	2	2.47	0.25	0	nan	nan
1na7A	3	2.72	0.07	0	nan	nan	0	nan	nan
7catB	8	2.54	0.19	2	2.47	0.05	0	nan	nan
1qduA	5	2.09	1.06	1	2.26	0	0	nan	nan
3mupA	6	2.66	0.12	1	2.64	0	0	nan	nan
2m1wA	5	2.61	0.11	0	nan	nan	0	nan	nan
4j93A	4	2.5	0.39	1	2.31	0	0	nan	nan
4pf3A	4	2.5	0.13	1	2.48	0	0	nan	nan
3zzxB	0	nan	nan	0	nan	nan	0	nan	nan
2kecA	4	2.5	0.18	1	2.23	0	0	nan	nan
6q21D	4	2.45	0.26	0	nan	nan	0	nan	nan
2aa2A	2	2.75	0.11	0	nan	nan	0	nan	nan
2jcwA	6	2.68	0.24	1	2.35	0	0	nan	nan
1r6qA	8	2.23	0.5	0	nan	nan	0	nan	nan
2geqA	4	2.38	0.29	1	2.86	0	0	nan	nan
1e96A	7	2.39	0.58	0	nan	nan	0	nan	nan
5g3jA	6	2.48	0.22	1	2.31	0	0	nan	nan
4jnkA	7	2.55	0.25	2	2.49	0.24	0	nan	nan
1b34A	4	2.49	0.13	1	2.29	0	0	nan	nan
1hx1B	12	2.28	0.34	4	2.3	0.18	0	nan	nan
1y7qA	12	2.57	0.19	3	2.6	0.12	0	nan	nan
1tubA	3	2.33	0.67	1	1.38	0	0	nan	nan
5cbxA	2	2.07	0.04	1	2.11	0	0	nan	nan
1nwqC	10	2.1	0.43	3	2.06	0.27	2	2.16	0.28
