

COMPUTATIONAL PROTEIN STRUCTURE PREDICTION USING DEEP
LEARNING

A Dissertation

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

by

ZHAOYU LI

Professor Yi Shang, Dissertation Advisor

May 2020

The undersigned, appointed by the dean of the Graduate School, have examined the dissertation entitled

COMPUTATIONAL PROTEIN STRUCTURE PREDICTION USING DEEP
LEARNING

presented by Zhaoyu Li,

a candidate for the degree of Doctor of Philosophy

and hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Yi Shang

Dr. Dong Xu

Dr. Jianlin Cheng

Dr. Ioan Kosztin

DEDICATION

To my wife Xian, my parents, and my son Ethan.

ACKNOWLEDGEMENTS

I would like to take the opportunity to thank the following people who have supported me through my PhD program. Without the support of them, this thesis would not have been possible.

First, I would like to thank my advisor, Dr. Yi Shang. He has been my advisor since my master's study. He opened the door of research for me, showed me instructions, guided me, supported me, encouraged me, and inspired me whenever I need help. It is a great pleasure to work with Dr. Shang together. I would also like to thank my committee members, Dr. Dong Xu, Dr. Jianlin Cheng, and Dr. Ioan Kosztin. They have been giving me constructive suggestions and advices to my dissertation and helping me solve problems in my research. Their support is essential, and I appreciate their time and efforts very much.

I would also like to thank other lab mates. I am glad that I have known them, and I really enjoyed their company along the way. Special thanks to Junlin Wang, Wenbo Wang, Chao Fang, and Son Nguyen, who were always there for me.

Last but not least, I would like to thank my family, Xian Kang, Ethan Li, Fafen Li, Jinxiu Chang, and Min Li. They provided me great supports through the whole course of my study and without their help I would not have been here to pursue my degrees.

This work was supported in part by National Institutes of Health grant R01GM100701. The high-performance computing infrastructure is supported by the National Science Foundation under grant number CNS-1429294.

TABLE OF CONTENTS

DEDICATION	3
ACKNOWLEDGEMENTS	ii
TABLE OF CONTENTS	iii
LIST OF TABLES	viii
LIST OF FIGURES	x
CHAPTER 1. INTRODUCTION.....	1
1.1 Protein and Protein Structure Prediction	1
1.2 Protein Loop Modeling.....	2
1.3 Protein Contact Map Prediction.....	3
1.4 Protein Contact Map Refinement	4
1.5 Contributions	4
1.6 Thesis Organization	6
CHAPTER 2. BACKGROUND AND RELATED WORK.....	8
2.1 Protein Distance Map and Multidimensional Scaling	8
2.2 Protein Loop Modeling.....	9
2.2.1 <i>Ab-Initio Methods</i>	9
2.2.2 <i>Template-Based Methods</i>	10
2.3 Protein Contact Map Prediction.....	11
2.3.1 <i>Statistical Methods</i>	11
2.3.2 <i>Co-evolution of Protein Residues</i>	12

2.3.3	<i>Direct Coupling Analysis (DCA)</i>	13
2.3.4	<i>Machine Learning Methods</i>	14
2.3.5	<i>Evaluation Metrics</i>	15
2.4	Predicted Contact Map Refinement	17
2.4.1	<i>Existing Methods</i>	17
2.4.2	<i>Evaluation Metrics</i>	18
2.5	Deep Neural Networks	18
2.5.1	<i>Convolutional Neural Network</i>	19
2.5.2	<i>Fully Convolutional Network (FCN)</i>	20
2.5.3	<i>Residual Neural Network (ResNet)</i>	21
2.5.4	<i>Generative Adversarial Network (GAN)</i>	22
CHAPTER 3. MUFOLD-LM: PROTEIN LOOP MODELING USING GENERATIVE		
ADVERSARIAL NETWORK		
3.1	Motivations	23
3.2	Problem Formulation	24
3.3	MUFOLD-LM System Architecture	25
3.4	Deep Neural Network Structure	28
3.4.1	<i>Generator Network</i>	29
3.4.2	<i>Adversarial Discriminator Network</i>	30
3.4.3	<i>Implementation and Training</i>	30
3.5	Evaluation	32
3.5.1	<i>Benchmark Dataset</i>	32

3.5.2 Results	33
3.6 Conclusion	36
CHAPTER 4. MUFOLD-CONTACT: PROTEIN RESIDUE-RESIDUE CONTACT	
MAP PREDICTION.....	38
4.1 Motivations	38
4.2 Problem Formulation	39
4.3 Input Features	39
4.3.1 Dataset	40
4.3.2 Features Introduction	43
4.4 Deep Neural Network Structure	47
4.4.1 Predicting Distance Directly	48
4.4.2 Predicting Contact Map Directly.....	51
4.4.3 Two-Stage Multi-branch Network.....	55
4.5 Evaluation Results	59
4.6 Application: Contact Guided Modeling.....	64
4.7 Summary.....	67
CHAPTER 5. TPCREF: PREDICTED CONTACT MAP REFINEMENT	
5.1 Motivations	69
5.2 Problem Formulation	71
5.3 Overall Architecture	71
5.4 Refinement Process	72
5.4.1 Template Selection	72

5.4.2	<i>Contact Map Prediction Using an Existing Method</i>	73
5.4.3	<i>Template Contact Map Filters Generation</i>	73
5.4.4	<i>Predicted Contact Map Refinement</i>	74
5.5	Evaluation Results	75
5.5.1	<i>Dataset</i>	75
5.5.2	<i>Results</i>	75
5.6	Summary	78
CHAPTER 6. MUFOLD PLATFORM DEVELOPMENT		79
6.1	Introduction.....	79
6.2	Contributions	79
6.3	System Architecture Overview	80
6.4	Database Preparation	80
6.5	Template Searching and Selection.....	81
6.5.1	<i>Template Selection</i>	82
6.5.2	<i>Template Representation</i>	84
6.6	Loop Modeling	84
6.6.1	<i>Loop Modeling by Fragment Alignments Searching</i>	85
6.6.2	<i>Loop Modeling by Shortest Path</i>	87
6.7	Model Generation	87
6.7.1	<i>Method 1: Fully Extended Model Generation</i>	87
6.7.2	<i>Method 2: Distance Matrix Based Model Generation</i>	88
6.8	MUFOLD Modules Design	91

6.9 Model Quality Assessment	92
6.10 Web Services Development.....	93
6.10.1 Web Server Architecture	93
6.10.2 MUFOLD SSW - Secondary Structure and Supersecondary Structure Prediction Server	94
6.10.3 MUFOLD 3D Structure Prediction Server.....	98
6.10.4 MUFOLD Contact Prediction Server.....	99
6.10.5 Server Usefulness.....	100
6.11 Summary.....	101
CHAPTER 7. SUMMARY AND FUTURE WORK.....	103
7.1 Summary.....	103
7.2 Future Work.....	106
7.2.1 Loop Modeling.....	106
7.2.2 Contact Map Prediction.....	106
7.2.3 Predicted Contact Map Refinement.....	110
7.2.4 A Unified Web Portal for All Our Tools	111
VITA.....	112
REFERENCE	113

LIST OF TABLES

Table 2.1 Different methods for inferring DCA.....	14
Table 3.1 Loop modeling results on 8-Res benchmark.	33
Table 3.2 Loop modeling results on 12-Res benchmark.	34
Table 3.3 Loop modeling visualization of 2 loop region structures from Exp_noGAN and Exp_GAN.	35
Table 4.1 Overview of all features used in our contact map prediction program.	40
Table 4.2 Database PDB25 generation parameters	41
Table 4.3 GDT-TS results between our model and RaptorX on CASP11 testing set.....	50
Table 4.4 Target list used for contact map evaluation.....	59
Table 4.5 The long-range top L/5 precision for each milestone and the corresponding ranks.....	60
Table 4.6 Contact prediction precision comparison with other tools using CASP13 targets.....	62
Table 4.7 Contact prediction precision comparison with other tools using NewPDB50_Testing targets.	63
Table 4.8 CASP13 results on contact map guided modeling.	66
Table 5.1 TPCref refined contact prediction precision comparison with other tools using NewPDB50_Testing targets.	76
Table 5.2 Shannon entropy of contact map predictions by MUFOLD-Contact and existing methods.....	77

Table 5.3 Shannon entropy of contact map predictions by TPCref and existing methods.
.....78

Table 7.1 The average accuracy for contact derived from best predicted model in the pool
in CASP13. 109

LIST OF FIGURES

Figure 1.1 An example of loop modeling problem. The loop region can be filled in in different ways.	3
Figure 1.2 An example of contact map and distance map of protein 3A35-A.	4
Figure 2.1 A protein 2D distance map of $C\alpha$ atoms (left) and the corresponding 3D structure (right). They can be converted to each other.	9
Figure 2.2 An example of co-evolution in biology.	13
Figure 2.3 Example of contact map prediction output and the evaluation scores.	17
Figure 2.4 Convolutional operation with a 3 by 3 kernel.	20
Figure 2.5 An example of using fully convolutional operation to get a pixel classification output.	21
Figure 2.6 Residual neural network building block structure.	22
Figure 3.1 Example of image inpainting problem.	23
Figure 3.2 The complete 2D representation of a protein (left) and the incomplete representation with a loop (right).	24
Figure 3.3 The flowchart of our loop modeling method.	26
Figure 3.4 Network structure for protein loop modeling using GAN.	28
Figure 3.5 Visualization of intermediates predicted results for case 1ALC. (The first four distance maps are at training steps 1, 100, 200, 300 respectively, the fifth to the eleventh distance maps are at training steps from 600 to 2400 with 300 incremental steps. The last one in red box is the ground truth).	36

Figure 4.1 Example of image semantic segmentation.	38
Figure 4.2 Length distribution of all samples in original dataset version 2.	42
Figure 4.3 Length distribution of all samples in filtered dataset version 2.	43
Figure 4.4 An example of PSSM output file.	44
Figure 4.5 Assignment of eight-state Shape Strings as eight clustered regions with specific boundaries on the Ramachandran plot.	46
Figure 4.6 Network structure for distance map prediction.	49
Figure 4.7 An example testing case 2G1U of our distance map prediction model.	51
Figure 4.8 The features flowchart in network.	53
Figure 4.9 Network structure to predict binary contact directly.....	54
Figure 4.10 Network structure of the two-stages multi-branch network.....	56
Figure 4.11 Network structure for distance prediction in the first stage in the two-stages multi-branch network.....	57
Figure 4.12 Dilated kernels and the receptive fields.	58
Figure 4.13 The graph for long-range top L/5 precision for each milestone and the corresponding ranks.....	61
Figure 4.14 Chart of contact prediction precision.	62
Figure 4.15 Flowchart of contact map guided modeling.....	65
Figure 4.16 CASP13 results on contact map guided modeling.....	66
Figure 5.1 An table to illustrate the idea of user-based collaborative filtering.	70
Figure 5.2 The flowchart of TPCref.....	71

Figure 5.3 Comparison between the TPCref contact map filter generation and the traditional user-based collaborative filtering.	74
Figure 5.4 Improvements after applying TPCref	76
Figure 5.5 Long L/5 results comparison of NewPDB50_Testing targets.	77
Figure 6.1 System architecture of MUFOLD pipeline.	80
Figure 6.2 Illustration of overlap regions in distance matrix in loop modeling.	85
Figure 6.3 Illustration of loop modeling process by patching a gap in the distance matrix.	86
Figure 6.4 Illustration of how fully extended modeling method works.	88
Figure 6.5 Process of iterative MDS in distance matrix based modeling.	90
Figure 6.6 The process of fixing mirror case.	90
Figure 6.7 MUFOLD modules design.....	92
Figure 6.8 Web server architecture.	94
Figure 6.9 MUFOLD_SSW server job submission page.	96
Figure 6.10 The email notifications when a job is submitted and finished.	97
Figure 6.11 MUFOLD_SSW server job running status page.....	97
Figure 6.12 MUFOLD_SSW server job result page.	98
Figure 6.13 World map of MUFOLD server visitors.	100
Figure 6.14 Recent feedback received from our server user.	101
Figure 7.1 The idea of using PatchGAN in distance map prediction network.	108

COMPUTATIONAL PROTEIN STRUCTURE PREDICTION USING DEEP LEARNING

Zhaoyu Li

Dr. Yi Shang, Dissertation Advisor

ABSTRACT

Protein structure prediction is of great importance in bioinformatics and computational biology. Over the past 30 years, many machine learning methods have been developed for this problem in homology-based and ab-initio approaches. Recently, deep learning has been successfully applied and has outperformed previous methods. Deep learning methods could effectively handle high dimensional feature inputs in modeling the complex mapping from protein primary amino acid sequences to protein 2-D or 3-D structures. In this dissertation, new deep learning methods and deep learning networks have been proposed for three problems in protein structure prediction: loop modeling, contact map prediction, and contact map refinement. They have been implemented in the state-of-the-art MUFOLD software and obtained significant performance improvement.

The goal of loop modeling is to predict the conformation of a relatively short stretch of protein backbone. A new method based on Generative Adversarial Network (GAN), called MUFOLD-LM, is proposed. The protein 3-D structure can be represented using the 2-D distance map of C_α atoms. The missing region in the structure will be a missing region in the distance map correspondingly. Our network uses the Generator Network to fill in the missing regions in the distance map based on the context, and the Discriminator Network will take both the predicted complete distance map and the ground truth as input to

distinguish between them. The method utilizes both the features and context of the missing loop region to make better prediction of the 3-D structure of the loop region. In experiments using commonly used benchmark datasets 8-Res and 12-Res, MUFOLD-LM outperformed previous methods significantly, up to 43.9% and 4.13% in RMSD, respectively. To the best of our knowledge, it is the first successful GAN application in protein structure prediction.

The goal of contact map prediction is to predict whether the distance between two C_β atoms (C_α for Glycine) in a protein falls within a certain threshold. It can help to determine the global structure of a protein in order to assist the 3D modeling process. In this work, a new two-stage multi-branch neural network based on Fully Convolutional Network and Dilated Residual Network, called MUFOLD_Contact, is proposed. It formulates the problem as a pixel-wise regression and classification problem. The first stage predicts distance maps for short-, medium-, and long-range residue pairs. The second stage takes the predicted distances from stage 1 along with other features as input to predict a binary contact map. The method utilizes the distance distribution information in the feature set to improve the binary prediction results. In experiments using CASP13 targets, the new method outperformed single stage networks and is comparable with the best existing tools.

In addition to predicting contact directly using deep neural networks, a new method, called TPCref (Template Prediction Correction refinement), is proposed to refine and improve the prediction results of a contact predictor using protein templates. Based on the idea of collaborative filtering from recommendation system, TPCref first finds multiple template sequences based on the target sequence and uses the templates' structures and the templates' predicted contact map generated by a contact predictor to form a target contact-

map filter using the idea of collaborative filtering. Then the contact-map filter is used to refine the predicted contact map. In experimental results using recently released PDB proteins, TPCref significantly improved the contact prediction results of existing predictors, improving MUFOLD_Contact, MetaPSICOV, and CCMPred by 5.0%, 12.8%, and 37.2%, respectively.

The proposed new methods have been implemented in MUFOLD, a comprehensive platform for protein structure prediction. It provides a rich set of functions, including database generation, secondary and supersecondary structure prediction, beta-turn and gamma-turn prediction, contact map prediction and refinement, protein 3D structure prediction, loop modeling, model quality assessment, and model refinement. In this work, a new modularized MUFOLD pipeline has been designed and developed. Each module is decoupled from each other and provides standard communication protocol interfaces for other programs to call. The modularization provides the capability to easily integrate new algorithms and tools to have a fast iteration during research. In addition, a new web portal for MUFOLD has been designed and implemented to provide online services or APIs of our tools to the community.

CHAPTER 1. INTRODUCTION

1.1 Protein and Protein Structure Prediction

In biology, proteins are large biomolecules consisting of chains of amino acid residues. Proteins perform a vast array of functions with different structures. Understanding the three-dimensional structure of a protein is very important and helpful to understand the protein functions. In 1972, Anfinsen and his colleagues received the Nobel Prize for Anfinsen's dogma: A protein's native structure is uniquely determined by its amino acid sequence (Anfinsen, 1973). From that time, it became the biggest challenge in structural bioinformatics to predict a protein's structure given only its amino acid sequence (Samish, Bourne, & Najmanovich, 2015).

There are different levels of structural organization of a protein: protein sequence, secondary structure, tertiary structure, and quaternary structure. The protein sequence is a sequence of amino acid residues with each amino acid represented using a letter. The secondary structure is a form of local segments of a protein. It can be classified into alpha helix, beta sheet, and coil. A tertiary structure of a protein is the three-dimensional structure of the complete protein. A quaternary structure is a complex of multiple proteins.

The problem of protein structure prediction and modeling has been actively applied in many areas such as drug design, protein classification, gene function annotation, and immunotherapy. The traditional experimental methods such as X-ray crystallography or Nuclear Magnetic Resonance (NMR) can give accurate structure of a protein but they could be very time-consuming and expensive (Johnson, Srinivasan, Sowdhamini, & Blundell, 1994).

1.2 Protein Loop Modeling

For the word “loop”, there are two different meanings. One is referred to as the segments that are not α -helix or β -sheet structures in secondary structure. Another one is referred to as the regions with insertions or deletions in the target sequence or templates. This research focuses on the second meaning in this research because in the homology protein modeling problem, the template may have gaps, which is a region that the amino acid has no atomic coordinates. In this case, it is important to reconstruct these missing regions for protein functions and dynamics studies (Ginalski, 2006). It is a small-scale 3D structure prediction since we know the protein sequence of the missing segments and we try to predict the structure from its sequence and its surrounding known structures. The quality of the complete protein prediction can be improved if we can fill in those missing regions very well. The prediction of those missing regions is called the loop modeling problem (Levefelt & Lundh, 2006). The following Figure 1.1 shows an example of loop modeling. The model on the left side is the initial model with a missing region, and the three models on the right side are the complete models with the missing region filled in by three different structures.

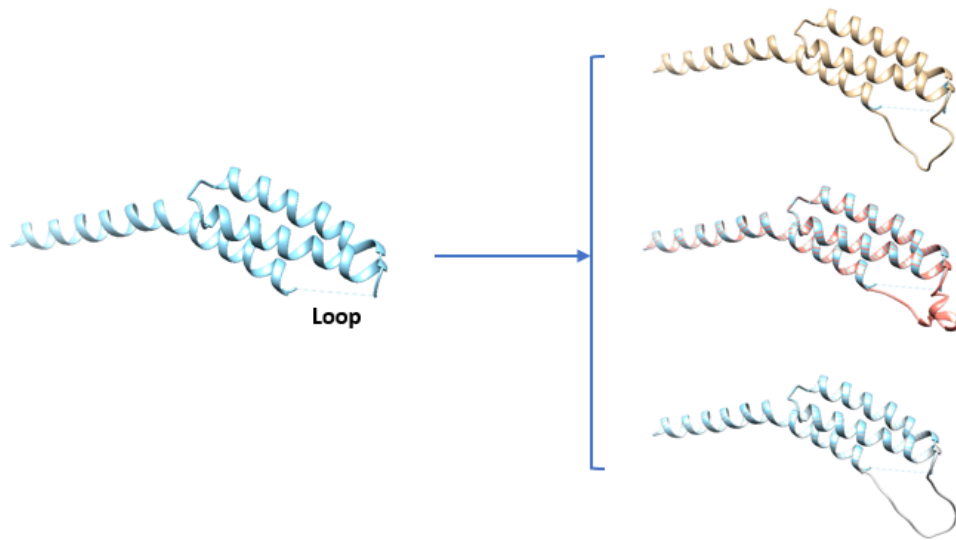


Figure 1.1 An example of loop modeling problem. The loop region can be filled in in different ways.

1.3 Protein Contact Map Prediction

A protein contact map is a binary matrix representing the presence or absence of spatial contact between all pairs of amino acid residues of a protein. For a protein with length L , the contact map of it is a binary $L \times L$ matrix C , in which each element C_{ij} is defined by:

$$C_{ij} = f(x) = \begin{cases} 0, & \text{if } D < \text{threshold} \\ 1, & \text{otherwise} \end{cases}$$

where D is the Euclidean distance between C_{β} atoms (C_{α} for glycine) of residue i and j .

The threshold is usually 8 angstroms.

Similar to contact map, distance map is used in our research. Distance map is like contact map, where the former contains real distance values and the latter contains whether the distance is within a threshold. The following Figure 1.2 shows an example of the contact map (left) and the distance map (right) of protein 3A35-A. In this figure, the value -1 means the corresponding residues are missing in the PDB file.

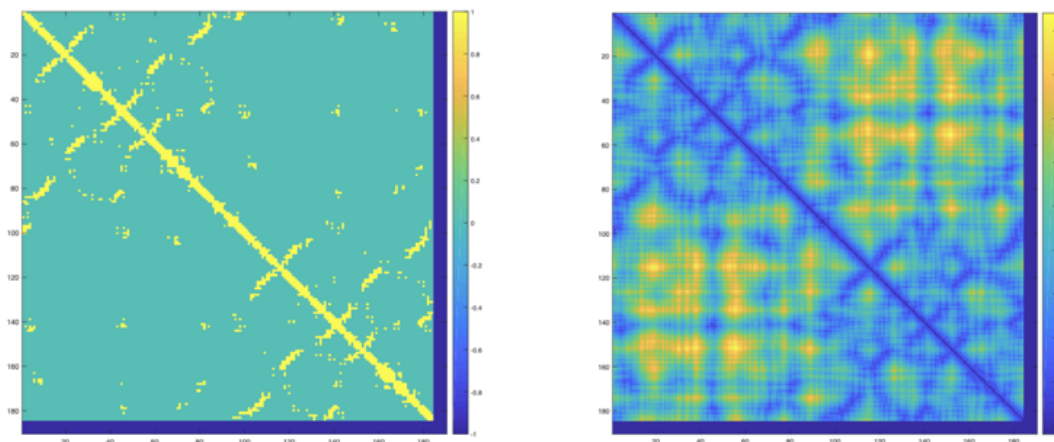


Figure 1.2 An example of contact map and distance map of protein 3A35-A.

Contact map prediction is very important because it can help to determine the 3D structure of the protein (Vendruscolo, Kussell, & Domany, 1997). Knowing if two residues are in contact or not gives us a basic topology structure of the protein and can reduce the conformation searching space. In addition, distance map is also very useful since by applying Multidimensional Scaling algorithm we can convert a distance map to 3D structure (Jingfen Zhang et al., 2010).

1.4 Protein Contact Map Refinement

State-of-the-art contact prediction methods are not yet capable of correctly predicting all contacts for a given amino acid sequence. In order to improving the quality of predicted contact maps, we can add a post process after the predictions are done. The post process will use more information that may be missing in the prediction stage to refine the predicted contact maps.

1.5 Contributions

This thesis makes the following contributions:

1. For protein loop modeling problem, a Generative Adversarial Network (GAN) that can utilize the context information of the missing region from its templates and predict its distance map is proposed. The predicted distance map is converted back to 3D structure using Multidimensional Scaling algorithm. In our experiments, the GAN can improve the quality of the predicted missing region than without using GAN. To the best of our knowledge, it is the first successful GAN application in protein structure prediction.
2. For contact map prediction problem, a new two-stages multi-branch network based on fully convolutional neural network and residual network has been proposed. The first stage predicts the distance maps and the second stage predicts contact map. Extensive feature engineering and experiments have been done to understand the problem and to find better configurations of the network.
3. For contact map refinement problem, a new method, called TPCref (Template Prediction Correction refinement), is proposed to refine and improve the prediction results of a contact map predictor using protein templates. Based on the idea of collaborative filtering from recommendation system, TPCref creates a contact filter from the templates' prediction results and the templates' native structure. In our background evaluation, comparable methods for refining contact map predictions were not found.
4. The new MUFOLD protein structure prediction platform has been designed and implemented. It was refactored and developed using Objected Oriented Programming language and was divided into several modules. Each module is decoupled with each other and communicated with each other using our own well-

defined protocols. New tools or algorithms can be easily integrated and tested. Currently, our MUFOLD platform contains tools for protein structure prediction, secondary structure prediction, supersecondary structure prediction (Psi-phi angles, beta-turn and gamma-turn), and contact map prediction.

5. A web services system for our tool has been designed and implemented. One general backend system was developed to manage jobs submitted by user on the server side. Multiple frontend web portals were developed for each tool in MUFOLD. The web portal has been used widely in our community and is constantly updated to provide the best services.

1.6 Thesis Organization

This thesis is organized into the following sections:

1. Chapter 1 describes the introduction of computational biology and an overview of problems that this work focuses on.
2. Chapter 2 describes the background and related work of protein loop modeling and protein contact map prediction. The basic background of deep learning and the applications of deep learning in computational biology are also reviewed.
3. Chapter 3 introduces our tailored Generative Adversarial Network (GAN) for protein loop modeling. The proposed the GAN transforms the protein loop modeling problem into an image inpainting problem and outperforms all the previous predictors.
4. Chapter 4 introduces our deep neural network for protein contact prediction. The network consists of 4 networks in total and they are divided into two stages. The

stage 1 focus on predicting distance map and the stage 2 focus on predicting contact map.

5. Chapter 5 introduces our tool, TPCref, for protein contact map refinement. It explains how the idea of collaborative filtering is applied to the contact map refinement problem to utilize the knowledge of the classification results from predictions of template sequences.
6. Chapter 6 introduces the development of our comprehensive protein prediction platform MUFOLD. It is an end-to-end platform that can predict the 3D structure from the protein sequence. It has four main modules: the database generation, template searching and selection, model generation, and model selection. This chapter covers the system architecture overview and the details of each module. This chapter also introduces the development of our web services. Three online tools are provided to the community: the secondary structure and supersecondary structure prediction server, the 3D structure prediction server, and the contact map prediction server.
7. Chapter 7 summarizes all the previous work and describe the future work.

CHAPTER 2. BACKGROUND AND RELATED WORK

2.1 Protein Distance Map and Multidimensional Scaling

Usually the representation of the structure of a protein is using 3D coordinates for each atom in amino acid. Those coordinates can uniquely define the spatial locations of atoms in amino acid. However, this representation is orientation dependent. When the structure rotates, the values of coordinates change. A protein can have infinite such representations while all of them are identical after superimposing. In this work, another representation of the protein structure, which is a 2D distance map of C_α atoms, is used. If the length of the protein sequence is N , the distance map will be a N by N matrix, in which each value is a real Euclidean distance of C_α atoms of two amino acids. This representation is orientation independent and easier to handle in deep neural networks.

Multidimensional Scaling (MDS) can be used to restore the protein 3D structure from its distance map (Jingfen Zhang et al., 2010). MDS is an efficient method for solving the graph realization problem. It can find a placement of points from another multidimensional space in the current multidimensional space, where the distances between points resemble the original dissimilarities. In our case, MDS can restore the 3D structure of C_α while preserving the distance constraints in the 2D space, as shown in the following figure. Then PULCHRA (Rotkiewicz & Skolnick, 2008) is used to build the full atom structure from the C_α backbone structure.

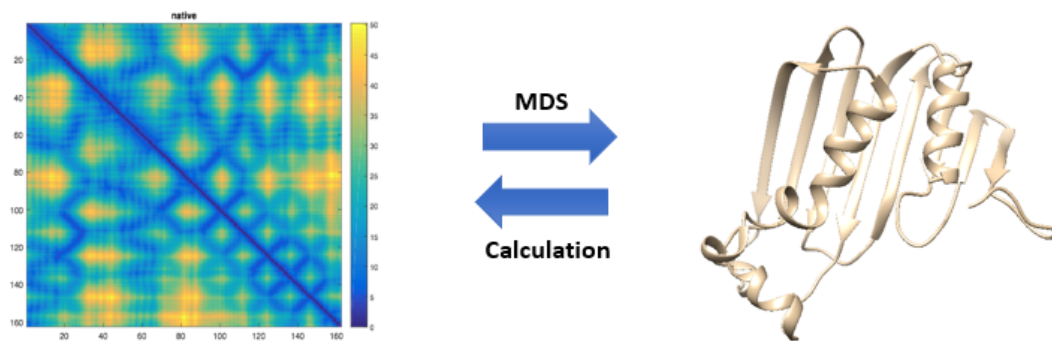


Figure 2.1 A protein 2D distance map of C_{α} atoms (left) and the corresponding 3D structure (right). They can be converted to each other.

2.2 Protein Loop Modeling

In general, there are two main strategies for the loop modeling problem: *ab initio* method and template-based method.

2.2.1 Ab-Initio Methods

The *ab initio* method is also regarded as a mini protein folding problem (Fiser, Do, & Sali, 2000). It generally contains initial sampling and loop conformation selection. It first generates conformations by some statistical methods under geometry constraints and fit the loop conformation into the gap by loop closure algorithms. Then use some selection algorithm in terms of minimization of the energy functions to select the final conformations.

ModLoop from MODELLER (Fiser et al., 2000) uses this method in their program. First, they build a straight line in the gap as the initial conformation of the loop, then use conjugate gradient minimization and simulated annealing to get the conformation with the lowest energy. The energy function is a sum of simple restraints in terms of distances and angles. Similar to ModLoop, another program Loopy (Xiang, Soto, & Honig, 2002) first

generates loop conformations by sampling torsion angle pairs. Then it uses random tweak loop closure algorithm to close the loop. Later, RAPPER (de Bakker, DePristo, Burke, & Blundell, 2003) program generates the loop conformation by combining fragments. The fragments are sampled by residue specific ϕ/ψ angles from one end of the loop towards the other end. Then they use anchor goodness-of-fit, SCWRL clash energy, and Samudrala-Moult potentials for final selections. Similarly, multiscale modeling method (Olson, Feig, & Brooks, 2008) using lattice-based low-resolution models and all-atom simulation is proposed to generate initial loop conformations. Then a physical energy-based scoring function is used to do selection. RCD+ (Lopez-Blanco, Canosa-Valls, Li, & Chacon, 2016) program uses the random coordinate descent (Chys & Chacon, 2013) algorithm to generate initial loop conformations, then the conformations are ranked by a distance-orientation dependent energy filter. Top ranked loops are refined with the Rosetta energy function.

2.2.2 Template-Based Methods

This method is also called database or knowledge based method. It finds existing loops in database, like Loops in Protein (LIP) (Michalsky, Goede, & Preissner, 2003) or Loop in Membrane Proteins (LIMP) (Hildebrand et al., 2009) databases, or the self-generated database, by superimposing the stem region of the gap and then selects ones with low RMSD or sequence similarity. SuperLooper (Hildebrand et al., 2009) is a program that searches LIP and LIMP databases to find loop candidates. Then all the candidates are scored based on sequence criteria and the RMSD of overlap regions. A more advanced database search algorithm in FREAD (Choi & Deane, 2010) uses four main filters in database search phase: anchor C_α separations, sequence similarities, statistical energy

function, and anchor RMSD. It proves that the sequence similarities filter can dramatically improve the results.

Some other tools use the combination of ab-initio and template based methods or use some other hybrid methods, for instance, NGK (Stein & Kortemme, 2013), Galaxy PS1 (Park & Seok, 2012) and Galaxy PS2 (Park, Lee, Heo, & Seok, 2014). Among those methods, NGK performs sampling for the loop structure based on the idea of combining intensification of torsion and parameter annealing strategies. Both Galaxy PS1 and Galaxy PS2 are loop refinement methods that starts with an inaccurate loop structure. The energy of Galaxy PS1 is optimized for application to the refinement of template-based models, while Galaxy PS2 is developed for higher performance for the near native models.

2.3 Protein Contact Map Prediction

During the protein structure prediction, if we know the contact map then we can derive the structure topology by following the constraints in the contact map (Vendruscolo et al., 1997). Therefore, researchers have been working the contact map prediction method for a long time. There are several time periods with different methods as the mainstream and the performance of prediction is growing continuously.

2.3.1 Statistical Methods

At the early stage, researchers were trying to apply pure statistical methods to predict contact map. There are two types of statistical methods: the local statistical methods and global statistical methods.

The local statistical methods treat each pair of two residues in the sequence is statistically independent from each other. It is not widely used because it suffers from the

transitive effects between multiple residue pairs so that it cannot distinguish between direct and indirect correlation signals (Weigt, White, Szurmant, Hoch, & Hwa, 2009). The global statistical methods, on the other hand, consider all other residue pairs while predicting one pair.

The global statistical model that is commonly used to describe this joint probability distribution is the Pott's model (F. Y. Wu, 1982), and different types of methods were proposed to infer the parameters for the Pott's model. Other than the traditional log-likelihood maximization method, other direct coupling analysis (DCA) based methods were proposed as well. For example, message passing DCA (mpDCA) in 2009 (Weigt et al., 2009), pseudo-likelihood approximations (plmDCA) in 2011 (Ekeberg, Lövkvist, Lan, Weigt, & Aurell, 2013), mean-field inversion approximations (mfDCA) in 2011 (Morcos et al., 2011), and GaussianDCA in 2014 (Baldassi et al., 2014), etc. So far, pseudo-likelihood is proved to be the most successful approximation for contact prediction, and it is widely implemented in different tools, such as the GREMLIN (Kamisetty, Ovchinnikov, & Baker, 2013) and CCMPred (Seemayer, Gruber, & Soding, 2014), or plmDCA, etc.

2.3.2 Co-evolution of Protein Residues

In biology, co-evolution means two species affect each other's evolution. For example, in the following Figure 2.2, the crab and snail are co-evolved because when the crab has more powerful claws, the snail will have much thicker shells that can protect itself from the crab.

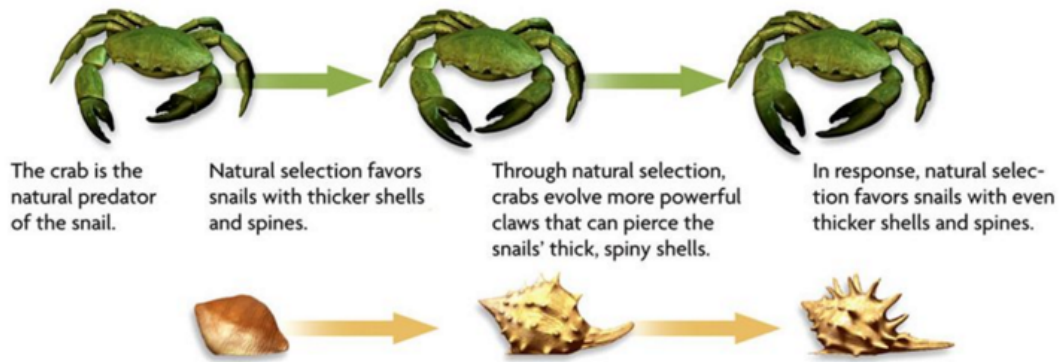


Figure 2.2 An example of co-evolution in biology.

The same concept applies to protein residues. In proteins, random mutations occur in residues over time. When a residue mutates, to preserve protein's function or structure, there is a compensatory residue mutates elsewhere in the protein, and this phenomenon is called correlated mutation. The key idea is those co-evolved positions are more likely to be in contact because they are closer to each other in 3D space (Godzik & Sander, 1989).

2.3.3 Direct Coupling Analysis (DCA)

The invention of DCA in 2009 was a breakthrough to predict contacts from sequences (Weigt et al., 2009). Different inference methods for DCA have been proposed and the following Table 2.1 is a summary of main DCA methods.

Table 2.1 Different methods for inferring DCA.

DCA methods	Tools	Year
Message passing algorithm (mpDCA)	-	2009
Original pseudo-likelihood approximations	GREMLIN	2011
Naive mean-field inversion approximation (mfDCA)	EVfold FreeContact	2011
Sparse inverse covariance estimation	PSICOV	2012
Improved pseudo-likelihood method	CCMPred plmDCA GREMLIN	2013
Related approach to mfDCA and PSICOV	GaussianDCA	2014

The mpDCA is computational expensive and only applicable to small proteins. It is not widely used for contact prediction now, but it is a breakthrough from traditional methods to DCA based methods. Later, mfDCA was proposed and it was around 100 times faster than mpDCA. Pseudo-likelihood-based methods were proposed later, and they were significantly outperformed previous methods.

One disadvantage of DCA methods is that they require very large multiple sequence alignments to provide accurate contact predictions. This problem has been overcome by refining the initial DCA prediction with deep learning methods.

2.3.4 Machine Learning Methods

Multiple machine learning based methods have been proposed to learn the mapping between features and contact map, such as SVMCon (Cheng & Baldi, 2007) and SVM-SEQ (S. Wu & Zhang, 2008) that are using support vector machine (SVM), PhyCMap (Z. Wang & Xu, 2013) that are using random forests. In recent years, many neural network or deep learning based methods have been proposed and they have improved the performance

significantly, such as NNCon (Tegge, Wang, Eickholt, & Cheng, 2009), DNCon (Eickholt & Cheng, 2012), and RaptorX (Sheng Wang, Sun, Li, Zhang, & Xu, 2016), etc.

Another type of predictors has been proposed recently that are combining multiple other predictors' results. This type of predictor is called meta predictor. Since each single predictor has its own algorithm and advantage, if they are combined then we can take advantage of each one's features. There are several meta predictors. The PconsC (Skwark, Abdel-Rehim, & Elofsson, 2013) combines sequence features and predictions from PSICOV and plmDCA. The metaPSICOV (David T Jones, Singh, Kosciolk, & Tetchner, 2014) and RaptorX all combine sequence features and predictions from PSICOV, mfDCA, and CCMPred, etc. The EPSILON-CP (Stahl, Schneider, & Brock, 2017) and NeBcon (B. He, Mortuza, Wang, Shen, & Zhang, 2017) combine five and eight other predictors' results, respectively.

2.3.5 Evaluation Metrics

There are many kinds of evaluation metrics depending on the purpose of using contact map. In this work the evaluation metrics in CASP community (Monastyrskyy, D'Andrea, Fidelis, Tramontano, & Kryshtafovych, 2014, 2016; Monastyrskyy, Fidelis, Tramontano, & Kryshtafovych, 2011) is followed. According to CASP definition, a contact means the distance of C_β atoms (C_α for glycine) of two residues is less than 8 Å.

The main performance metric is the mean precision, which is the ratio of the number of true contacts to the top scoring predicted contacts, as the following equation shows:

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

where the TP is the true positive contacts and FP is the false positive contacts.

Since the number of residue pairs is very large and it is not necessary to evaluate all of them. Therefore, all residue pairs are divided into three categories depending on the residue distance, i.e. the separation between two residues in the protein sequence. The definition of different contact ranges is shown in the following equation and we usually evaluate three ranges separately.

$$\text{Contact Range} = \begin{cases} \textit{short}, & \textit{separation} < 6 \\ \textit{medium}, & 6 \leq \textit{separation} \leq 23 \\ \textit{long}, & \textit{separation} \geq 24 \end{cases}$$

In each contact range, we select the top N scoring predicted contacts to evaluate and N depends on the length of the protein. Generally, we choose N as the following values: $L/2$, $L/5$, and 10 .

There is another evaluation metric called F1 score used by CASP for ranking by default. F1 score is the harmonic mean of precision and recall in range of 0 and 1. It is defined by the following equation:

$$F1 = 2 \times \frac{\textit{Precision} \times \textit{Recall}}{\textit{Precision} + \textit{Recall}}$$

The following Figure 2.3 shows an example of sample contact prediction output and the overview of all scores to be evaluated. In the sample predictions output, we can see in each line there are five values delimited by space. The first two values are the residue ID in the sequence. The next two values are the distance definition of in contact. Here it means two residues are in contact if the Euclidean distance of C_{β} atoms of these two residues is between 0 and 8 angstroms. The last value is the predicted probability of these two residues are in contact. We use the probability as the score to rank the predictions.

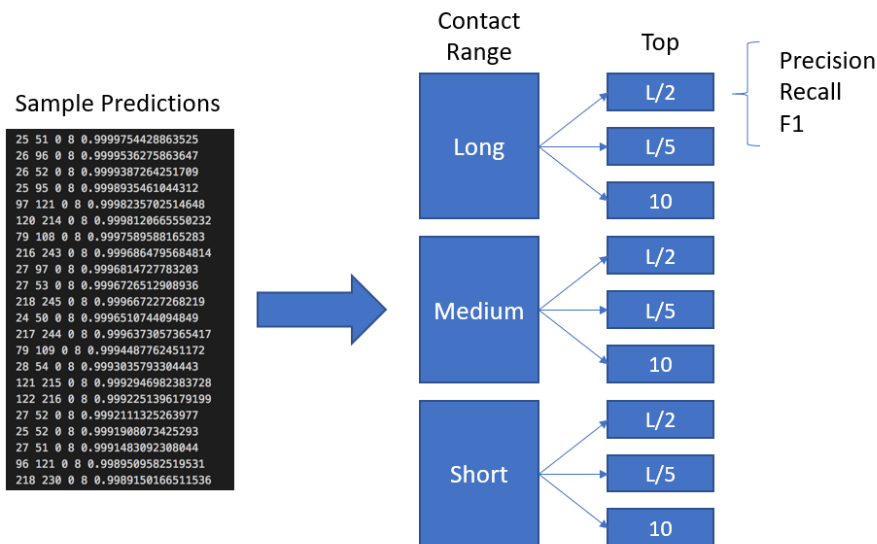


Figure 2.3 Example of contact map prediction output and the evaluation scores.

2.4 Predicted Contact Map Refinement

2.4.1 Existing Methods

In 3D protein structure prediction, refinement is a very common process since the prediction is not accurate enough to be comparable to the experimentally determined native structures. The goal of protein structure refinement is to make a starting structure closer to its native structure. There are many existing protein structure refinement tools such as GalaxyRefine (Heo, Park, & Seok, 2013) and 3DRefine (Bhattacharya, Nowotny, Cao, & Cheng, 2016) etc.

However, to the best of our knowledge, there is no such work for the contact map prediction. Most of contact map predictors take sequence as input and give a contact map prediction directly. We are the first one to propose a tool that can be used for any other existing contact map predictors to do post process refinement.

2.4.2 Evaluation Metrics

The evaluation metrics for refined contact map are the same as traditional contact map as described in Chapter 2.3.5. Additionally, the diversity of predicted and refined contact maps was considered, as it has been shown to be an important consideration for protein structure prediction (B. He et al., 2017; Kinch, Li, Monastyrskyy, Kryshtafovych, & Grishin, 2016). Diversity was evaluated as the Shannon entropy of the top-L predicted contacts (for all ranges together and for only long-range), with the contact map divided into a 10 by 10 grid (B. He et al., 2017), according to the following equation:

$$H = - \sum_{e=1}^{100} p_e \log_2 p_e$$

where H is the Shannon entropy, equal to a negative sum over each of the 100 contact map cells, and p_e is the fraction of top-L predicted contacts within the e -th cell which were correctly predicted (Y. Li, Hu, Zhang, Yu, & Zhang, 2019).

2.5 Deep Neural Networks

In recently years, deep learning has made a huge impact on Computer Science and achieved unprecedented performance on many Machine Learning problems, such as image classification (Krizhevsky, 2012), speech recognition (Hinton et al., 2012), and natural language processing (Collobert & Weston, 2008).

The development of Graphic Processing Unit (GPU) makes it possible to train very large neural networks faster. Generally, each deep neural network contains an input layer, hidden layers, and an output layer. The number of hidden layers can be very large so that the deep neural network can be trained with many parameters using a large amount of data. Those

hidden layers can extract the complex abstract representations of data automatically rather than human-designed representations. Many advanced deep neural network architectures have been proposed, such as Convolutional Neural Network (Krizhevsky, 2012), Recurrent Neural Network (Mikolov, Karafiát, Burget, Cernocký, & Khudanpur, 2010), and Residual Neural Network (K. He, Zhang, Ren, & Sun, 2015). They have shown great performance improvement in various fields. For bioinformatics, deep learning has also been applied to many problems and has achieved some good results, such as secondary structure prediction (S. Wang, Peng, Ma, & Xu, 2016), loop modeling (Z. Li, Nguyen, Xu, & Shang, 2017), quality assessment (J. Wang, Li, & Shang, 2017) and protein contact map prediction (Sheng Wang et al., 2016).

2.5.1 Convolutional Neural Network

In the fields of multiple data types processing, such as two-dimensional image processing and classification, the convolutional neural network (CNN) is an important architecture and has achieved great performance on ImageNet competition. It is inspired by the organization of the animal visual cortex. The basic architecture consists of the convolutional layer, followed by the pooling layer, and at the end some fully connected layers and the output layer. The convolutional layer contains multiple feature maps to extract features from the input data. Local connectivity and parameter sharing are common methods to reduce the number of parameters. The pooling layer is to perform the non-linear down sampling. The most common method is max pooling. After several convolutional layers and pooling layers, there is a fully connected layers to do the final decision. The architecture of convolutional neural network allows the network to learn

more and more abstract features in a higher level. The following shows the basic operation of convolution with a 3×3 kernel.

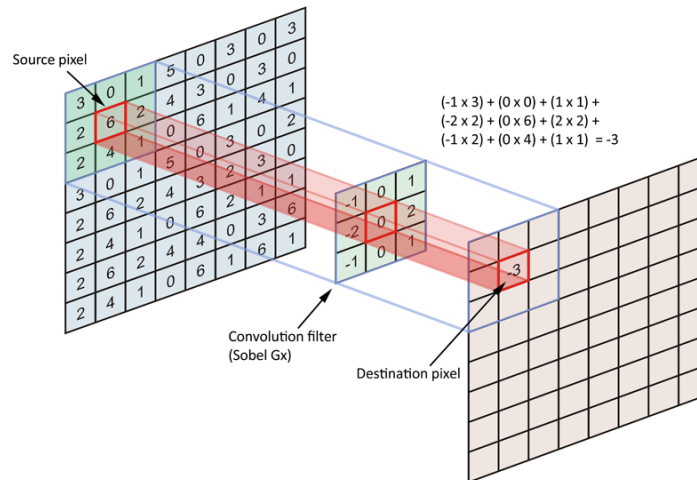


Figure 2.4 Convolutional operation with a 3 by 3 kernel.

2.5.2 Fully Convolutional Network (FCN)

The fully convolutional network (FCN) was proposed in this paper (Shelhamer, Long, & Darrell) for image semantic segmentation. It can take input of arbitrary size and produce the correspondingly sized output. In traditional CNN classification network, there are fully connected layers at the end. The fully connected layers map the feature maps from convolution layer to a fixed length feature vector, for example, the AlexNet (Krizhevsky, 2012). Therefore, the input size must be fixed because the output length is fixed. To solve this problem, FCN was proposed, and it transforms the fully-connected layers into convolution layers with kernel size 1×1 since the convolution operation doesn't care about the input size and a 1×1 kernel can do an operation similar to fully connected layer for each pixel in the feature depth axis. The following Figure 2.5 shows an example of applying 1×1 kernel to get a same-sized output.

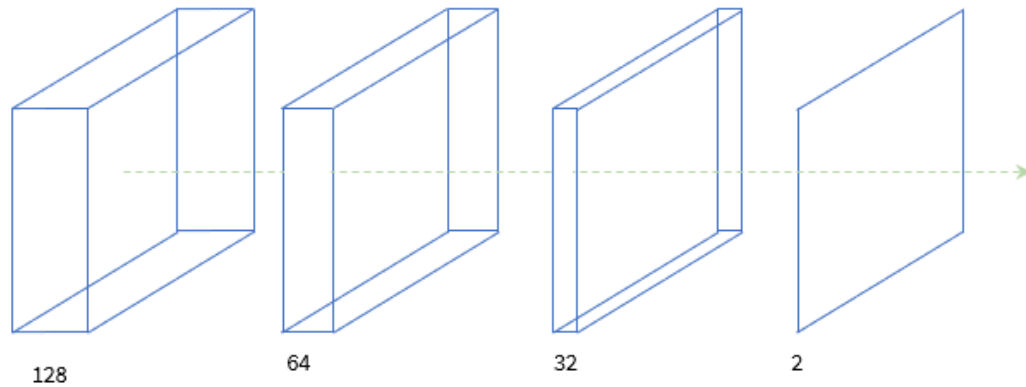


Figure 2.5 An example of using fully convolutional operation to get a pixel classification output.

The network can be trained end-to-end using arbitrary size of input and in one forward propagation all the pixels can be predicted.

2.5.3 Residual Neural Network (ResNet)

The residual neural network is introduced recently in the ImageNet competition the results were quite impressive (K. He et al., 2015). The basic structure of this network architecture is the building block, in which there could be some other layers. If the input of a building block is X , then the output of this block is $X + X'$, in which X' is the non-linear transformation of X . A function f indicating the difference between the input and output of the building block is defined, then f is called residual function. This feature of the network will give it the ability to learn something different from what the input has already encoded. Also, this network will handle the vanishing gradient problem very well. The basic structure of a building block is shown in the following figure.

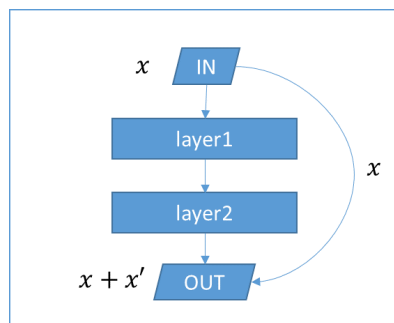


Figure 2.6 Residual neural network building block structure.

The residual neural network is used in this work for the following reasons. First, the protein prediction problem is very complicated and there will be a very large number of parameters to be trained. In order to do so, very deep neural network is needed. Traditional network architectures are not able to handle this, while residual neural network, which is inspired originally by the highway network, can handle this very well. Second, the residual neural network is very similar to the recurrent neural network so that it is a good model to process sequential data. Third, this model has been tested in the most recent work and they have achieved great performance for prediction of contact map. We believe by making improvements based on the state-of-the-art work we can make a difference as well.

2.5.4 Generative Adversarial Network (GAN)

Generative Adversarial Network (GAN) was proposed recently as a new type of deep neural network (Goodfellow et al., 2014), in which two models are trained simultaneously: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G , and also provides gradients to model G . It is like a two-player game that G tries to generate more realistic results to fool D while D tries not to be fooled by G .

CHAPTER 3. MUFOLD-LM: PROTEIN LOOP MODELING USING GENERATIVE ADVERSARIAL NETWORK

3.1 Motivations

Our method is inspired by the image inpainting problem, which is that given an image with a missing region, we need to construct the missing region to make the whole image look as real as possible (Yeh, Chen, Lim, Hasegawa-Johnson, & Do, 2016), as the following Figure 3.1 shows.

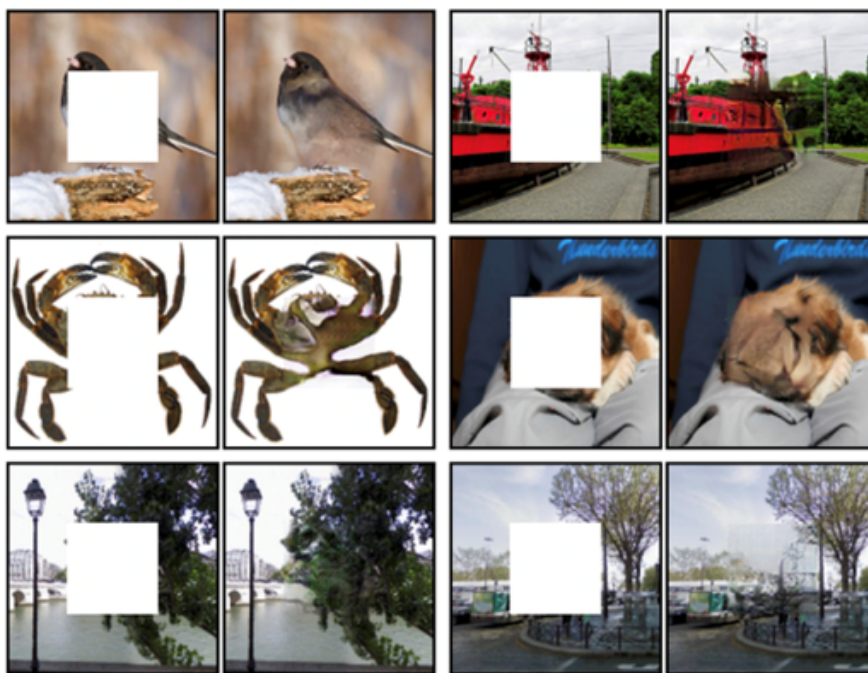


Figure 3.1 Example of image inpainting problem.

Several machine learning approaches have been proposed to deal with this problem. Deep Convolutional Generative Adversarial Network (DCGAN) is recently used to do semantic image inpainting (Yeh et al., 2016). They proposed two loss functions: the perceptual and contextual losses. The perceptual loss ensures a perceptually realistic output

image, and the contextual loss preserves similarity between the input corrupted image and the recovered image.

The image inpainting problem is like our loop modeling problem. When a protein has missing region, the corresponding distance map will have a missing region as well, like what the following Figure 3.2 shows. The missing region is highly dependent on the existing region following some specific patterns. We will let the deep neural network to learn those underlying patterns, i.e. the context, and predict the missing region based on what has been learned.

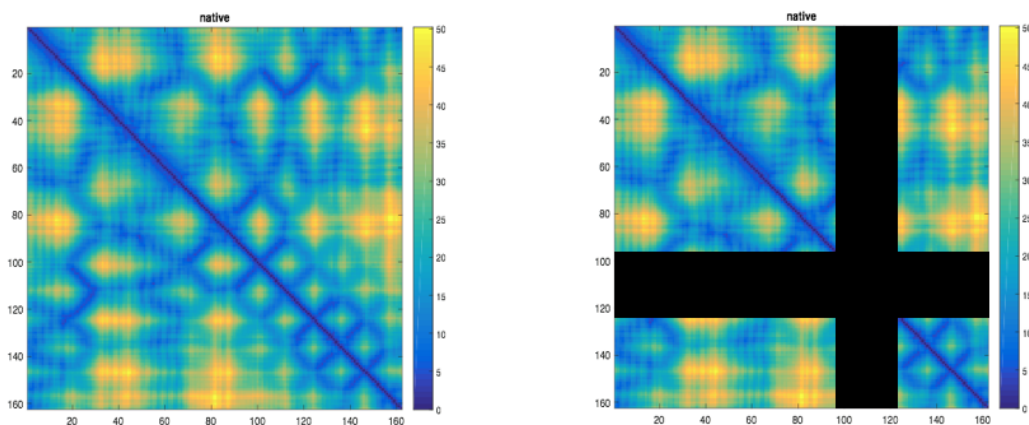


Figure 3.2 The complete 2D representation of a protein (left) and the incomplete representation with a loop (right).

3.2 Problem Formulation

The loop modeling problem is addressed as follows. Give a protein sequence S , in which M is a continuous segment and is it considered as the missing region. Our goal is to give the sequence S and the true coordinates of $S - M$ as the input and to predict the coordinates of the missing region M as the output. The M' is used as the notation for the predicted missing region.

The evaluation metric is the root mean square deviation (RMSD) between M and M' , and the objective is to make it minimal. The RMSD value is calculated using the coordinates of the corresponding main chain atoms (N , $C\alpha$, C and O) between M and M' , as shown in the following equation. Those atoms are generally the representative atoms of a protein and can be used to generate the full atom model.

$$RMSD_{loop} = \sqrt{\frac{1}{|M|} \sum_{i=1}^{|M|} \|M - M'\|^2}$$

Both M and M' need to be superimposed before calculation of RMSD.

3.3 MUFOLD-LM System Architecture

The MUFOLD_LM method can be divided into two parts: the template pool generation including subsequence extraction and alignment searching, and the neural network training and prediction. The term “target” is used to represent each protein with loop to be modeled. The following figure shows the overall system flowchart.

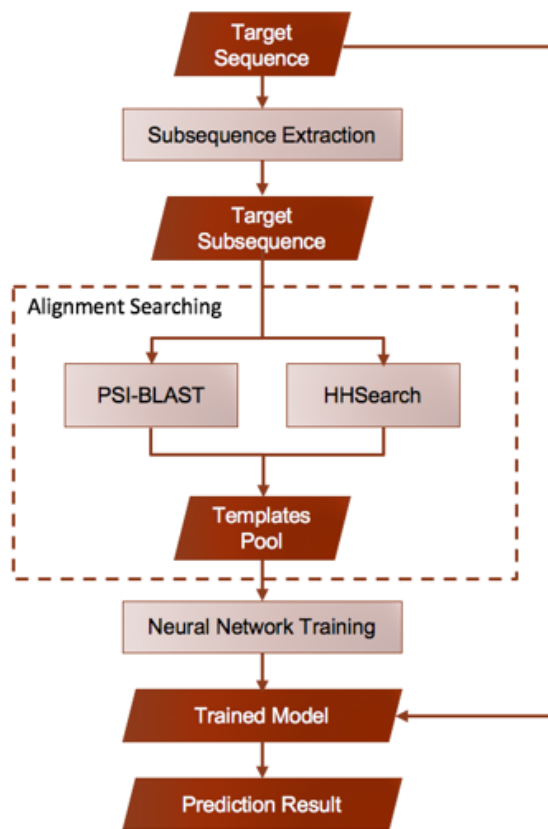


Figure 3.3 The flowchart of our loop modeling method.

Per-target Training and Predicting

MUFOLD_LM method trains one model for each target and then do prediction based on this target’s templates pool. Instead of picking the best patch as the template of the loop, it tries to learn the context of the loop region from a pool of candidate patches and predict the real loop region based on the learned context using deep learning techniques. It is still template based, but it has a more systematic way to better utilize information of many templates than previous methods. By using per-target training and predicting, the input features are prepared specifically for the input target and the information can be fully utilized in the network.

Subsequence Extraction

Given a target sequence, the loop region is located first and a subsequence with a length of 50 residues is extracted, in which the loop region will be in the middle. For example, if the loop length is 8, then there will be 21 residues on each side of this loop in the subsequence.

Alignment Searching

For each target subsequence, it is fed into alignment searching tools to get a pool of candidate templates. The alignment tools will search the Protein Database Bank (PDB) database and do a sequence similarity comparison to find similar templates. The results may contain incomplete structures, so a filter is applied to keep those with complete structures and 3D coordinates as the final templates. PSI-BLAST (Altschul et al., 1997) and HHSearch (Soding, 2005) are used for the alignment searching. In addition, all native-like templates in the results are excluded to make sure there is no ground-truth structures included. The templates do not have to cover a large portion of the target protein and they do not have to be statistically significant with small E-values. As long as they can cover the missing loop and its flanking region, even the templates with insignificant E-values can help.

Network Training

In order to fit the templates into a neural network, all the template candidates are converted into 50 by 50 distance maps. The distance map can be treated as an image in the following training. For each distance map the corresponding loop region is taken out and

the loop region is predicted based on the context of the rest of the distance map. More details are described in the following Deep Network Structure section.

Prediction

After the training, a model for the target subsequence is finished. When doing prediction, the incomplete distance map for the target subsequence will be the input, and the predicted loop region will be the output. Then MDS is used to restore the 3D structure of this 50-length segment and calculate the RMSD between the loop region’s structure and its true structure to see the performance.

3.4 Deep Neural Network Structure

MUFOLD_LM deep network contains two sub-networks: The Generator Network, and the Adversarial Discriminator Network. Two networks are trained simultaneously. The whole network structure is shown in the following figure. The term “native” is used to denote the ground truth structure.

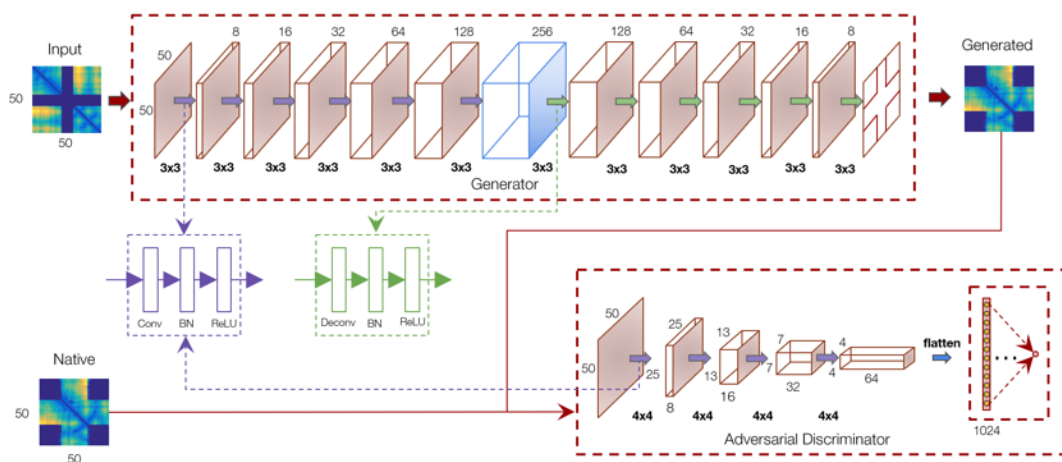


Figure 3.4 Network structure for protein loop modeling using GAN.

3.4.1 Generator Network

As shown in the above figure, the Generator Network consists of two parts: the convolutional half to capture the context information of the input into a latent representation in the middle (the blue box), and the de-convolutional half (the term “de-convolutional” sometimes has different meaning (Zeiler, Krishnan, Taylor, & Fergus, 2010)); in our paper, it means the transpose of the convolution) to reconstruct the missing region from the latent representation.

Convolution

Unlike the traditional convolutional networks, MUFOLD_LM network keeps the input dimension all the time during the convolutional process, i.e. only the number of channels changes, but the height and width are always the same with the input and output. It has been showed that keeping the dimension can give better performance since the latent representation in the middle can capture and keep more information (Masci, Meier, Cireşan, & Schmidhuber, 2011). There are 5 layers in this part and each layer contains a convolution operation, a Batch Normalization (BN) operation (Ioffe & Szegedy, 2015), and a rectified linear unit (ReLU) activation function (Glorot, Bordes, & Bengio, 2011).

De-convolution

This part of the network can be understood as a transpose convolution with learned filters. It is like an up-sampling but with the same height and width. The intuition behind this is the convolutional part captures the context information layer by layer and aggregates it into the latent representation, and the deconvolution will try to make use of the context information in the latent representation as much as possible to generate the missing region.

There are 5 layers in this part and each layer contains a de-convolution operation, a Batch Normalization (BN) operation, and a rectified linear unit (ReLU) activation function.

For the Generator Network, there is no pooling layer and fully connected layer. The final output layer is generating a 2D matrix, which can be treated as an image with only 1 color channel. The kernel size for all layers is 3 by 3 with stride 1, and the number of filters in each layer is noted in Fig. 4. The generated output in our actual implementation is the missing region plus an overlap region on each side. The overlap size we use is 5 so that if the input loop length is n , the predicted loop length will be $n + 10$. In this way, the generated results can have a better binding to its context.

3.4.2 Adversarial Discriminator Network

MUFOLD_LM's Adversarial Discriminator Network is a deep neural network with 4 convolutional layers and 1 fully connected layer. The input is a 50 by 50 distance map, and the output is the probability whether the input is generated by Generator Network or real. In this network, each convolution layer contains a convolution operation, a Batch Normalization (BN) operation, and a rectified linear unit (ReLU) activation function.

For the Adversarial Discriminator Network, the kernel size is 4 by 4 with stride 2 in order to have a larger receptive field with fewer number of layers. The number of filters in each layer is noted in the above figure

3.4.3 Implementation and Training

The whole network is trained to minimize the RMSD between the generated missing region and the ground truth to make the generated missing region more realistic. In order

to achieve this, two loss functions are defined: the reconstruction loss and the adversarial loss.

Reconstruction Loss

The reconstruction loss is to measure the similarity of the reconstructed missing region and the ground truth. The L2 distance between the predicted distance map of missing region and the distance map of ground truth is used. In the following equation, z is the input to the Generator Network (G), x is the ground truth.

$$Loss_{recon} = \|G(z) - x\|_2^2$$

Adversarial Loss

The adversarial loss is to pick the best distribution according to the training set. During the training, the Adversarial Discriminator Network (D) is trained to maximize $\log(D(x))$, where x is the ground truth, i.e. the probability of correct distinction between the generated result and the ground truth. In the meantime, the Generator Network (G) is trained to minimize $\log(1 - D(G(z)))$ in which z is the input to G:

$$\underset{G}{Min} \underset{D}{max} E_{x \in X} \log(D(x)) + E_{z \in Z} \log(1 - D(G(z)))$$

The adversarial loss is therefore defined as:

$$Loss_{adv} = \log(1 - D(G(z)))$$

The overall loss function is defined as:

$$Loss_{all} = Loss_{recon} + Loss_{adv}$$

The program is implemented using Tensorflow v1.0 (Abadi et al., 2016). For the training process, the Generative Network is trained at every step and the Adversarial Discriminator Network is trained at every 10 steps, which turns out to be the best

configuration. The Adam optimizer is used with the learning rate 0.0001 (Kingma & Ba, 2015). Early stopping is used to determine when to stop the training. The number of templates varies from 250 as the minimum to 617 as the maximum, with 415 on average. The template candidate dataset for each target is randomly divided into the training set and the validation set with the ratio of 9:1.

3.5 Evaluation

3.5.1 Benchmark Dataset

The benchmark for loop modeling performance used in our experiment is a dataset including 40 backbone perturbed targets from this paper (Park et al., 2014). In this dataset, 20 targets have an 8-length missing region in the structure (8-Res benchmark), and another 20 targets have a 12-length missing region in the structure (12-Res benchmark). The rest of the structure in a protein is known. Totally there are 40 targets, but there are two targets, 2SGA from 8-Res benchmark and 1C5E from 12-Res benchmark, that we can't make the missing region in the middle in subsequence extraction process, therefore we exclude those two targets in our experiments.

Since MUFOLD_LM deep network contains a Generative Network and an Adversarial Network, the performance of the single Generative Network without the adversarial part is tested to see if the Adversarial Network can really help and how much it can help to improve the performance. The first experiment is called Exp_noGAN. Then the whole network with both networks trained simultaneously is tested, which is called Exp_GAN.

MUFOLD_LM's results are compared with other state-of-the-art loop modeling tools including NGK, Galaxy PS1, and Galaxy PS2.

3.5.2 Results

The results for 8-Res benchmark are shown in Table 1, the results for 12-Res benchmark are shown in Table 2. All experiments were executed 10 times and the average RMSD scores are reported. The unit for all numbers is Å.

Table 3.1 Loop modeling results on 8-Res benchmark.

PDB*	NGK	Galaxy PS1	Galaxy PS2	Exp_noGAN	Exp_GAN
135L	3.9	3.7	4.3	1.2	1.2
1ALC	1.3	1.4	1.4	0.3	0.3
1BTL	0.4	1.3	0.9	0.4	0.6
1CEX	2.1	2.0	1.8	0.7	1.1
1CLC	0.4	0.4	0.3	1.6	0.6
1DDT	3.7	2.0	1.5	1.9	1.5
1EZM	4.3	4.2	3.8	2.1	2.2
1HFC	0.7	1.0	0.9	0.5	0.9
1IAB	1.0	2.2	1.8	0.8	1.3
1IVD	2.7	3.6	2.2	1.5	1.7
1LST	1.2	1.1	1.1	1.5	1.2
1NAR	1.4	2.1	1.8	2.5	2.7
1OYC	1.1	1.6	1.7	0.7	0.7
1PRN	8.3	6.9	8.8	1.9	1.0
1SBP	0.9	0.8	0.8	1.1	1.0
1TML	1.1	1.1	0.6	2.0	2.1
2CMD	1.9	4.0	2.3	1.0	0.9
2EXO	1.5	1.0	1.1	1.1	0.6
5P21	1.7	1.9	1.9	0.8	0.5
AVG.	2.08	2.23	2.05	1.24	1.15
STD. DEV	1.89	1.60	1.91	0.64	0.65

*For PDB 2SGA, the loop region can't be made in the middle of subsequence, so it is excluded here.

From the 8_Res benchmark results, we can see our method with GAN gets the best average result compared to other tools. It has an almost 44% improvement to the best of other tools, i.e. Galaxy PS2. In addition, our GAN method gets the smallest standard deviation, which means our network prediction is more stable without too many outliers. For the experiment without GAN, it can already beat other tools, and after adding GAN, the results get even better.

From the 12_Res benchmark results, we can see our method with GAN gets the best average result compared to other tools as well. The Galaxy PS2 has the result better than our method without GAN, but it has the disadvantage that it needs initial loop information to do refinement while our method can predict the missing region directly. Also, our method gets the smallest standard deviation in the same way as in 8-Res benchmark.

Table 3.2 Loop modeling results on 12-Res benchmark.

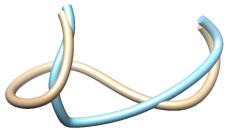

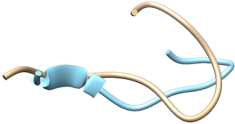
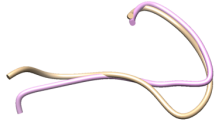
PDB*	NGK	Galaxy PS1	Galaxy PS2	Exp_noGAN	Exp_GAN
1A8D	3.7	4.5	3.1	2.8	2.7
1ARB	1.7	2.1	1.9	1.7	2.1
1BHE	1.7	2.3	3.5	2.8	3.2
1BN8	1.1	4.3	1.1	3.0	1.7
1CB0	0.9	5.7	0.9	1.1	1.7
1CNV	6.3	6.4	6.5	3.1	2.0
1CS6	1.1	1.7	1.6	3.8	4.0
1DQZ	7.5	1.5	3.3	2.0	2.6
1EXM	1.1	3.0	1.3	0.5	0.6
1F46	2.6	4.5	3.8	3.0	2.7
1I7P	1.9	2.8	1.7	1.6	1.3
1M3S	3.2	4.3	2.7	1.9	2.0
1MS9	1.8	1.8	1.8	1.0	1.4
1MY7	0.9	2.4	1.0	2.0	2.2
1OTH	0.8	1.1	0.9	2.4	2.2
1OYC	0.7	2.7	1.2	0.5	0.4
1QLW	6.0	2.5	2.9	4.3	2.9
1T1D	1.3	2.5	1.5	2.4	2.6
2PIA	0.7	4.5	0.7	3.5	1.4
AVG.	2.37	3.19	2.18	2.28	2.09
STD. DEV	2.07	1.48	1.43	1.07	0.88

*For PDB 1C5E, the loop region can't be made in the middle of subsequence, so we it is excluded here.

Two cases are further visualized: 1CLC from 8-Res benchmark, and 1BN8 from 12-Res benchmark, to see the differences between the predictions without GAN and the predictions with GAN as shown in Table 3. For all visualizations, the yellow model is the ground truth. The visualized models are superimposed already. From the visualization, it is observed that the GAN can make the prediction results smoother and more similar to the ground truth. For example, in the case of 1BN8, there is a small clash in the Exp_noGAN predicted

structure, which is fixed in the Exp_GAN predicted structure. In the case 1CLC, the predicted result without GAN is dramatically different, while in Exp_GAN the generated result is much better.

Table 3.3 Loop modeling visualization of 2 loop region structures from Exp_noGAN and Exp_GAN.

PDB	Exp_noGAN (Yellow is ground truth)	Exp_GAN (Yellow is ground truth)
1CLC (8_Res)	 <p data-bbox="672 751 795 779">RMSD: 1.6</p>	 <p data-bbox="1127 751 1243 779">RMSD: 0.6</p>
1BN8 (12_Res)	 <p data-bbox="672 974 795 1001">RMSD: 3.0</p>	 <p data-bbox="1127 974 1243 1001">RMSD: 1.7</p>

The intermediate results for case 1ALC with 8-length missing region are also visualized to see what is going on during the training. We use the model trained at each step to do a prediction on the input and to visualize the predicted distance map. In the following figure, the eleven distance maps are the intermediate results extracted from some specific training steps. The last distance map in red box is the ground truth. It can be observed that the missing region is filled in gradually. At the very beginning, it is like random values and has no relationship with the context. As the training goes, more and more values can be filled in together with the surrounding regions and the patterns of the filled in values get closer and closer to the ground truth.

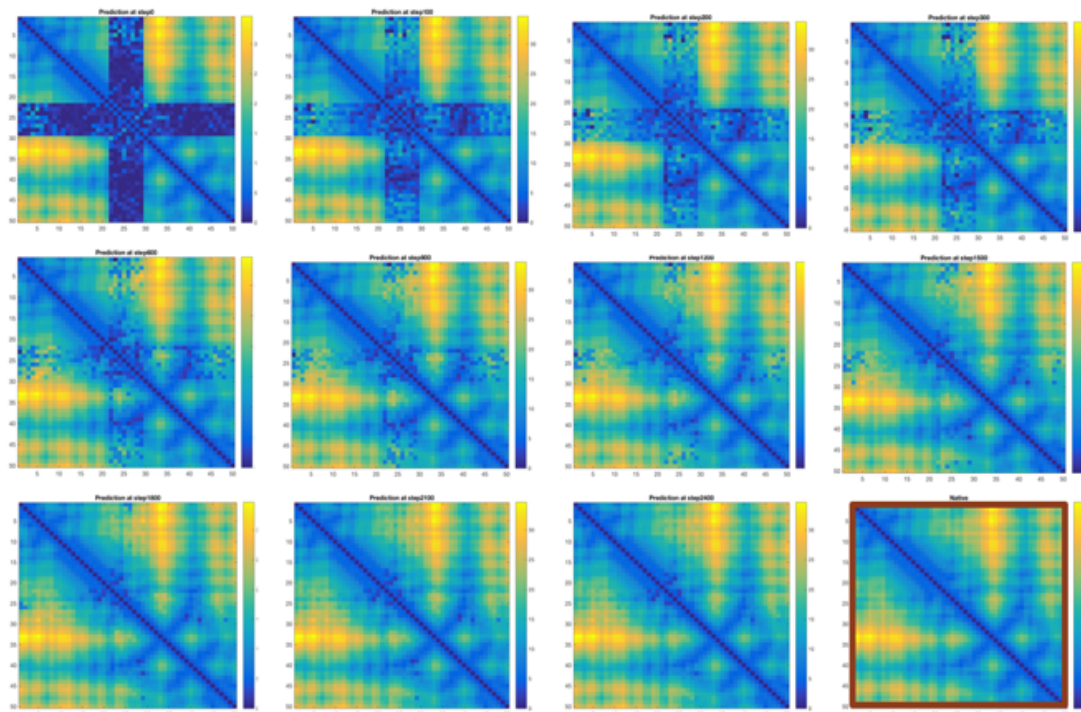


Figure 3.5 Visualization of intermediates predicted results for case 1ALC. (The first four distance maps are at training steps 1, 100, 200, 300 respectively, the fifth to the eleventh distance maps are at training steps from 600 to 2400 with 300 incremental steps. The last one in red box is the ground truth).

3.6 Conclusion

It is a novel approach for protein loop modeling using GAN. A Generative Network is used to produce prediction results, and an Adversarial Discriminator Network is used to distinguish if the result is generated by Generator or is the ground truth. Two networks are trained simultaneously. The network can learn the context of the loop region and predict the result based on that. The 3D protein structure is represented using 2D distance map in the neural network, which can be treated as image and is easier to handle. MDS is used to restore the 3D structure from 2D distance map. Experiments show that our overall method can achieve the best performance on both 8-Res benchmark and 12-Res loop benchmark.

The use of Adversarial Discriminator Network also gives better performance than using the single Generator Network.

As the first successful application of GAN in bioinformatics, MUFOLD_LM opens a door for many other GAN applications in analysis and prediction problems of biological sequences structures, such as protein contact prediction and 3D genome structure prediction.

CHAPTER 4. MUFOLD-CONTACT: PROTEIN RESIDUE-RESIDUE CONTACT MAP PREDICTION

4.1 Motivations

We know that a contact map is a binary matrix. If we treat it as a binary image, then each pixel in this image can be classified into two categories: in contact or not in contact. It is like a pixel-wise image classification. The most common pixel-wise image classification problem is image segmentation (Shelhamer et al., 2017), in which the input is an image and the output is a same size matrix with each pixel being classified into different categories. For example, if there are a desk and a chair in an image as input, then all pixels belong to the desk will be classified into one category and all the pixels belong to the chair will be classified into another category. The following Figure 4.1 shows the concept of image semantic segmentation.



Figure 4.1 Example of image semantic segmentation.

Inspired by image segmentation problem, our network is designed to do pixel-wise classification for protein contact map prediction.

Another problem we are facing is each protein has different length. For a traditional neural network, the input dimension should be fixed because if there is a fully connected layer at the end, the number of neurons for the fully connected layer is fixed. The convolution operation doesn't require the fixed dimension. Inspired by Fully Convolutional Network (FCN) (Shelhamer et al., 2017), we designed our network to be able to accept any length of protein as input and predict the corresponding size of contact map as output.

4.2 Problem Formulation

The protein residue-residue contact map prediction problem is addressed as follows. Given a protein sequence $S = \{R_1, R_2, \dots, R_L\}$. In the sequence, R_i represents the i th C_β atoms (C_α for glycine) of amino acid residue and the length of the sequence is L . Our goal is to predict a $L \times L$ matrix C , in which each C_{ij} represents whether the distance of R_i and R_j are within a threshold. If the distance is in the threshold, C_{ij} is 1, otherwise C_{ij} is 0. This matrix C is called the predicted contact map of protein sequence S .

4.3 Input Features

The features are all generated from the sequence only and can be categorized into two kinds: the 1D features and the 2D features. The 1D feature means the feature vector is for each amino acid in the sequence. The 2D feature means the feature vector is for each amino acid pair in the sequence. Assume the input sequence has length L , then the dimension for

1D feature is $L \times N$ in which N is the length of the feature vector, the dimension for 2D feature is $L \times L \times N$ in which N is the length of the feature vector.

In the following table, all the features that are used in our program are summarized, along with the dimension, the software used to generate the feature, and the database if needed in the feature generation.

Table 4.1 Overview of all features used in our contact map prediction program.

#	Feature Name	Dimension	Software	Database (if any)
1	PSSM	20	BLAST-2.7.1+	uniref90_042016
2	SS	3	Psipred_4.0	Profile from 1
3	SA	1	metaPSICOV 2.0.3	Profile from 1
4	Multiple Sequence Alignment (MSA)	-	HHsuite-2.0.16	uniprot20_2016_02
5	ALN Stats	$L \times L \times 3$	metaPSICOV 2.0.3 \rightarrow alnstats	MSA from 4
6	HHM Profile	30	Hhsuite-2.0.16 \rightarrow hhmake	MSA from 4
7	CCMPred	$L \times L$	CCMPred	MSA from 4
8	EVFold	$L \times L$	freecontact-1.0.21	MSA from 4
9	PSICOV	$L \times L$	PSICOV	MSA from 4
10	metaPSICOV	$L \times L$	metaPSICOV 2.0.3	From 2, 3, 5, 7, 8, 9
11	shapeString	8	frag1d	PSSM from 1
12	Raw Co-evolution Statistics	$L \times L \times 441$	cov21stats	MSA from 4
13	Physicochemical Features	5	Fixed values	-

4.3.1 Dataset

In our research process, two different datasets were tried as training and validation datasets. The first one is a smaller one that is a subset of RaptorX's training dataset (Sheng

Wang et al., 2016). This smaller dataset is called version 1. The second one is a larger dataset that is a subset of CullPDB (G. Wang & Dunbrack, 2003), and it is called version 2.

Dataset Version 1

In this version there are totally 3799 samples and it is a subset from RaptorX’s training dataset. The length of all samples is limited to larger than 50 and shorter than 300. All features are calculated by us with the following parameters.

Blast is manually executed using three iterations. The first two iterations are searching against the none-redundant sequence database to get a profile with e-value $1e-3$. The last iteration is searching against PDB sequence database with the previous profile as input and e-value $1.1e4$ to get as many as possible alignments with known structures. HHBlits is executed using three iterations with e-value $1e-3$ and is searching against UniProt20 database. The other tools are all using the default parameters.

Dataset Version 2

In this version, a subset of CullPDB is generated with the following parameters:

Table 4.2 Database PDB25 generation parameters

Database date	2018-09-18*
Maximum percentage identity	25%
Minimum resolution	0.0
Maximum resolution	3.0
Maximum R-value	1
Minimum chain length	40
Maximum chain length	700
Skip entries	non-X-ray, CA-only

* To be able to test on CASP13 targets, we later manually remove all entries that are released after 2018-05-01.

Since the maximum percentage identity is 25%, this dataset is called PDB25. Totally there are 13116 entries in this database. The following filtering are applied to the dataset:

1. Manually remove all entries that are released after 2018-05-01 so that the training dataset won't have any homology or native structure of CASP13 targets. In this way, the CASP13 targets can be used to compare the performance.
2. Remove all entries with Not A Number (NaN) values in the generated features or with missing features.

After the filters above, finally the version 2 dataset has 10896 entries and the following figure shows the length distribution of all samples.

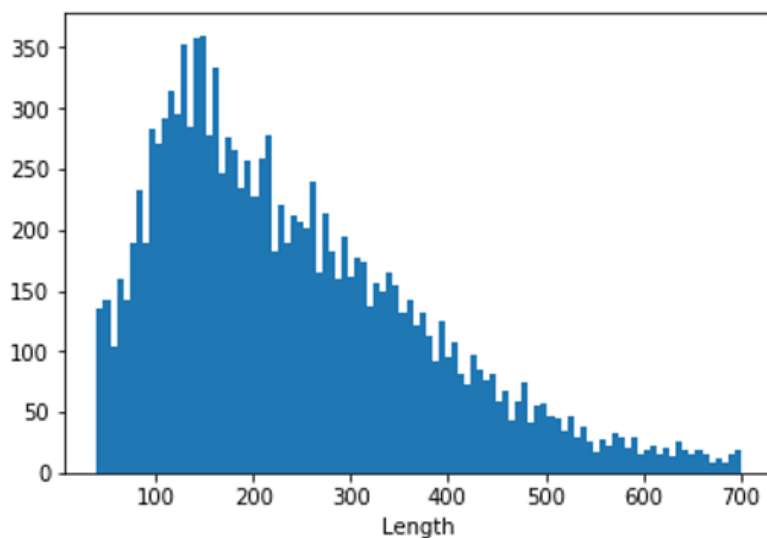


Figure 4.2 Length distribution of all samples in original dataset version 2.

During our experiments, it is shown that sometimes the training curve was fluctuated sharply on this dataset. After investigation on the fluctuation point, it is found to be caused by some training samples with too many missing values in it or the sample itself is too short. Some more filtering rules are applied:

1. Limit the length of training sample in range 50 and 500.
2. Calculate the percentage of missing residues in the protein sequence. If the percentage is equal or greater to 5%, the training sample will be removed.

After the above filters, there are 5250 samples left. The new length distribution is shown in the following:

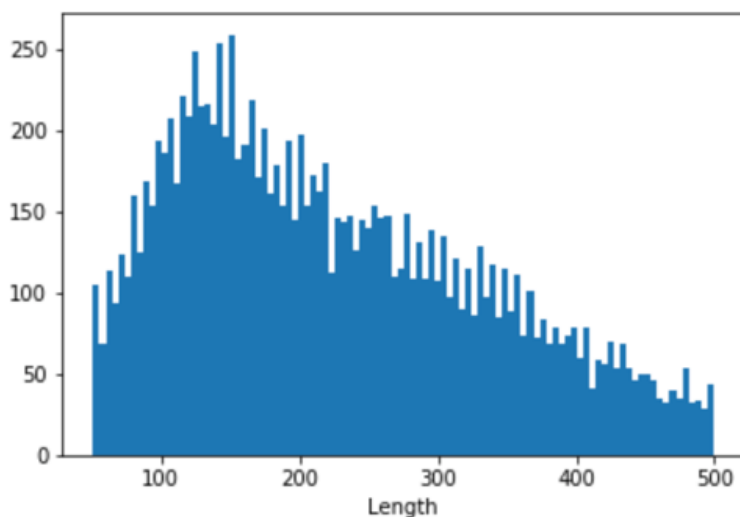


Figure 4.3 Length distribution of all samples in filtered dataset version 2.

4.3.2 Features Introduction

A brief introduction of every feature is given in this section.

1. Position-Specific Scoring Matrix (PSSM): Each amino acid has a length-20 feature vector. It is generated by BLAST based on the amino acid frequencies of every position in the multiple sequence alignment (D. T. Jones, 1999). Each score in the feature vector represents the frequency of substitution occurs in the multiple sequence alignment. All scores are scaled from -10 to 10 with positive values indicate more frequent substitution and negative values indicate less frequent substitution. The following figure shows a snippet of an example PSSM file.

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
1 S	0	-3	3	-2	-3	-2	-2	1	-2	-4	-4	-2	-3	-4	3	4	1	-5	-4	-3
2 H	-1	-2	3	-1	-4	3	1	-3	4	-4	-4	-1	-3	-4	2	-1	3	1	-1	-3
3 M	-2	1	-1	-3	-4	0	-2	-4	-4	0	0	-1	-2	-1	6	-1	-1	-5	-2	0
4 R	-2	6	-2	-3	-5	1	0	-5	1	-3	-5	1	-4	-5	2	-1	0	-5	-4	-5
5 A	1	1	-2	-1	-4	-1	0	-2	0	-5	-4	1	-4	-5	6	0	1	-5	-2	-4
6 P	-1	2	-2	-1	-2	0	-1	-3	2	-3	0	0	-2	-3	5	0	0	-1	1	-1
7 L	0	0	3	-2	-3	-1	-2	-3	0	0	2	1	-1	-2	0	-1	0	1	-2	0
8 D	-2	1	2	2	-2	0	-1	-3	-2	-3	-4	3	-3	-4	0	2	3	-3	-3	-3

Figure 4.4 An example of PSSM output file.

2. Secondary Structure (SS): Secondary structure is a small structure segment with specific patterns in a protein. It can be classified into eight classes or three classes. In our program we use the three-classes definition, i.e., helix (H), strand (E) and coil (C). PSIPred (D. T. Jones, 1999) is used to generate secondary structure prediction and use one-hot encoding as representation.
3. Solvent Accessibility (SA): Each protein has a 3D structure so in the space, some residues can be touched by water and some cannot. For those that can be touched by water are called exposed residues, and those cannot be touched by water are called buried residues. Solvent accessibility indicates whether the residue is predicted as exposed or buried. Same as secondary structure, one-hot encoding is used as representation.
4. Multiple Sequence Alignment (MSA): This is an intermediate result in our feature generation. MSA is a set of amino acid sequence alignments and they are all like the query target sequences. It is generated by HHBlits from HHSuite toolkit and used to generate many other features (Remmert, Biegert, Hauser, & Söding, 2011).
5. Alignment Statistics: We use three alignment statistics, i.e. the mean contact potential, normalized mutual information and mutual information. Those features are for each amino acid pair and each statistic is an $L \times L \times 1$ matrix. Those features

are generated by tool 'alnstats' from metaPSICOV toolkit based on the multiple sequence alignment we generated before (D. T. Jones, Singh, Kosciolatek, & Tetchner, 2015).

6. Hidden Markov Model (HHM) Profile: It is a profile generated by 'hhmake' for profile-profile comparison in protein sequence searching (Remmert et al., 2011). Each amino acid has a length-30 feature vector.
7. CCMPred: It is a score matrix from Direct Coupling Analysis (DCA) on multiple sequence alignment (Seemayer et al., 2014). DCA is the basic and the most popular method for contact map prediction now. There are different inference methods for DCA and CCMPred score is generated based on pseudo-likelihood maximization method.
8. EVFold: This is another score matrix generated by an algorithm based on the mean-field approximation of DCA (mfDCA). The tool used for this feature is freecontact (Kaján, Hopf, Kalaš, Marks, & Rost, 2014).
9. PSICOV: This is another score matrix generated by sparse inverse covariance estimation (D. T. Jones, Buchan, Cozzetto, & Pontil, 2012).
10. metaPSICOV: This is a toolkit for contact map prediction by combining different coevolution methods (D. T. Jones et al., 2015). We use their output in stage 2 as one kind of score matrix.
11. ShapeString: This is a predicted eight states features based on the discrete state of dihedral angles (Zhou, Shu, & Hovmoller, 2010). The eight-state Shape Strings are defined as shown in the following figure and this feature is generated by tool Frag1D.

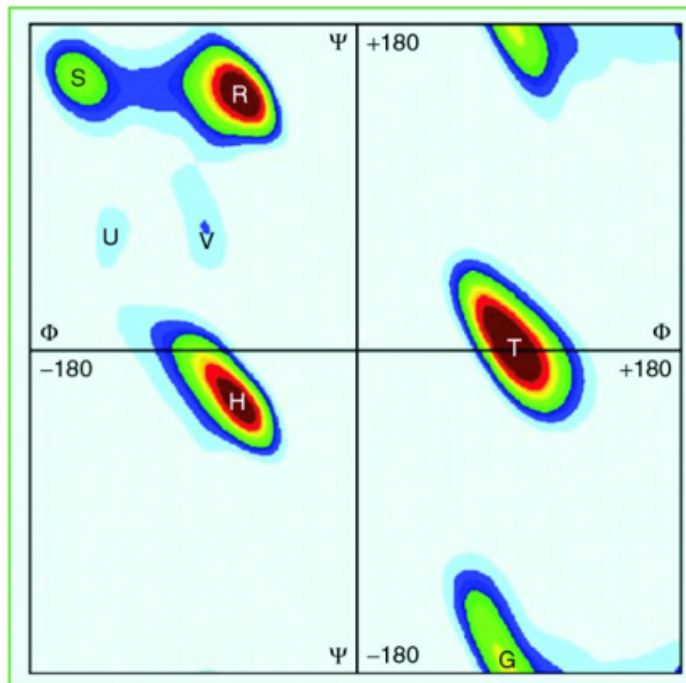


Figure 4.5 Assignment of eight-state Shape Strings as eight clustered regions with specific boundaries on the Ramachandran plot.

12. Raw Co-evolution Statistics: From above we use three different kinds of evolutionary coupling scores, i.e. CCMPred, PSICOV, and EVFold. All those scores are all inferred from Direct Coupling Analysis and are compressed to an $L \times L \times 1$ matrix. This raw score is the intermediate results during the inference. For each residue pair R_{ij} there is an 21×21 weight matrix with each weight represents the co-evolution statistic between 21 types of amino acid at position i and 21 types of amino acid at position j .
13. Physicochemical Features: For each amino acid there is length-5 fixed value feature vector which is pre-calculated based on the physical and chemical features of each amino acid type in nature. Because the values are always fixed if the amino acid type is determined therefore this is like a kind of sequence encoding method.

In order to utilize all the useful information from the features and make the model learning process more efficient, for the features that are real continuous values, all values are normalized to a range between -1 and 1. The experiments showed that after normalization the network converged better and faster, and among all normalization methods that have been tried, this min and max scaler normalization got the best performance.

4.4 Deep Neural Network Structure

Deep learning techniques are used for contact map prediction in order to utilize the information from multiple sequence alignments. The intuition behind this is among all the features, the most important and informative features will be selected by deep network automatically through the training process. Different network structures have different abilities. It is not possible to directly tell which network structure is the best one for protein contact map prediction until most of them have been tried. Some other important hyperparameters can affect the network performance drastically, such as the learning rate scheduler, the optimization method, or the data augmentation methods. All of those should be fine-tuned to gradually improve the model's performance.

Our development of the deep networks can be divided into to three stages. In the first stage, regression is tried for each residue pair to get a distance map prediction. The generative adversarial network is also tried to make the distance distribution more realistic. In the second stage, classification is tried for each residue pair. Each residue pair will be classified into either in contact or not in contact. In the third stage, the distance map prediction and binary contact map prediction are combined. The distance map can provide

more information of overall distribution of all residue pairs and that information is beneficial to do binary contact classification. Combining them together gives the best performance for now. In the following sections, the details of each stage are covered.

4.4.1 Predicting Distance Directly

Protein contact map reflects the physical distance of a protein. If the distance can be predicted, then it is easy to derive the contact map. Inspired by this idea, regression is done on the last layer.

Input Features

For this network, the following features are generated from the sequence as the input, and each sequence has the real distance map as the ground truth.

1. Features for each residue includes the Position-Specific Scoring Matrix (PSSM), secondary structure prediction, and solvent accessibility prediction, the dimension of those features are 20, 3, and 3, correspondingly. Therefore, for each residue, the feature vector length is 26.
2. Features for each residue pair includes the evolutionary coupling score from CCMPred, the mutual information, and the contact potential, the dimension of those features are all 1 for each. Therefore, for each residue pair, the feature vector length is 3 and for a protein sequence with length L , the feature dimension is $L \times L \times 3$.

Network Structure

Inspired by ResNet (K. He et al., 2015), the residual block is used to build the network and make it very deep. As shown in the following Figure 4.6, the input to the network has the dimension of $L \times L \times 55$. For each residue R_i in the sequence it has a length-26 feature

vector, then for each residue pair R_iR_j it has a length-52 feature vector by concatenating the feature vector of R_i and R_j , leading to a $L \times L \times 52$ feature input. After further concatenating with the $L \times L \times 3$ feature for residue pairs, the input to the network has the dimension of $L \times L \times 55$. The output of the network is an $L \times L \times 1$ distance map prediction, in which each pixel, i.e. residue pair, has a positive real value.

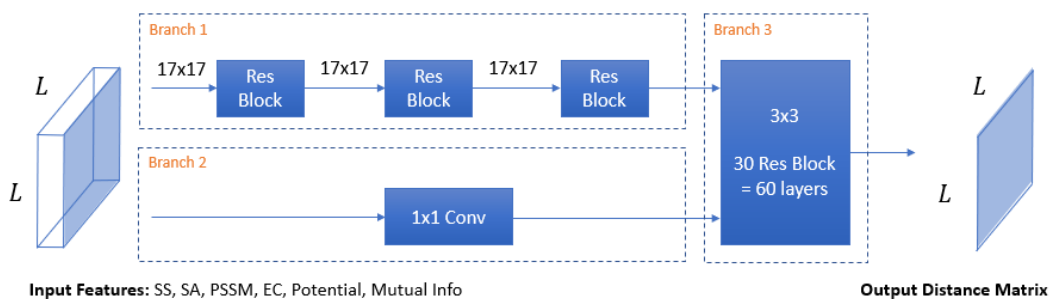


Figure 4.6 Network structure for distance map prediction.

In addition to residual blocks, the idea of fully convolutional network is applied so that the feature maps in each layer will always be $L \times L$ to retain the length of the protein.

The input features will go through a network branch with 3 residual blocks and each residual block has a kernel size of 17×17 . A large kernel size is used to capture the global features as much as possible since the long-range contact map prediction is more important and harder to predict. The output feature map size of this branch is $L \times L \times 64$. The input features will also go through another network branch with only a 1×1 convolution operation to make the output feature map size to $L \times L \times 64$. After concatenation, the feature size will be $L \times L \times 128$ and it will go through the third network branch. The third network branch is consisted of 30 residual blocks, i.e. 60 convolutional layers since each residual block has two convolutional layers. The kernel size of the third branch is 3×3

and the feature map size will be kept $L \times L \times 128$ until the last two layers with a reduction from 128 to 64 and further to 1 as in the final output.

The loss function used in training is mean square error (MSE). The optimizer is Adam with learning rate $1e-4$. Training batch size is 1, i.e. for each iteration there is only one training sample go through the forward propagation and to calculate the gradients to update the weights of the network. The main reason is each protein has different length. Although the convolutional operation is length independent, we can't process a batch of protein with various length at once. The activation function in this network was ReLU and all batch normalization layers were removed since our batch size is 1. The training set is using the dataset version 1 as described in previous sections.

The targets in CASP11 were used as testing set (Liu, Wang, Eickholt, & Wang, 2016). The state-of-the-art at that time was RaptorX-contact (Sheng Wang et al., 2016) and their intermediate results are also distance map. To compare the performance, we first applied multidimensional scaling on the distance map to get the 3D structure (Jingfen Zhang et al., 2010). The Global Distance Test Total Score (GDT-TS) was used to evaluate the model quality, which is a similarity score between the predicted model and the native model (Zemla, 2003). The GDT-TS is in range 0 to 1 and the higher score, the better quality. The following Table 4.3 shows the average GDT-TS between our method and RaptorX.

Table 4.3 GDT-TS results between our model and RaptorX on CASP11 testing set

	RaptorX-contact	Ours
Average GDT-TS	0.3331	0.2917

There were some good predictions in the testing set, for example, protein 2G1U had a predicted distance map that was like the native by simply eyeball observation. If converted

to 3D structures they had a GDT-TS score 0.6643, and after superimposed two structures, we can see the general structure was successfully predicted and generated. This example is shown in the following Figure 4.7.

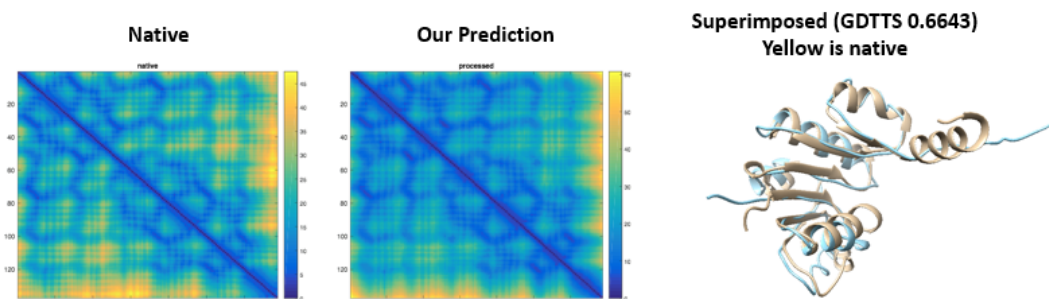


Figure 4.7 An example testing case 2GIU of our distance map prediction model.

Although there were several cases with good prediction results, the overall results are not good enough at that time since our network was preliminary. It proved that the idea of predicting distance map could work if the model was further fine-tuned. This model was improved later for the following versions of our networks.

4.4.2 Predicting Contact Map Directly

Although distance map can be used to reconstruct 3D structure using multidimensional scaling, there are multiple constraints in the distance map. The deep neural network can predict the distance value for each residue pair but cannot guarantee the physical distance constraints. The MDS fails if the distance map is disturbed. Therefore, the contact map derived from the distance map may not accurate and may cause clashes in the protein. It is the same situation in our previous experiment that some distance map prediction fails the MDS algorithm.

Based on the above observation, another network to do pixel classification is designed and developed, i.e. each residue pair will be classified to either 0 or 1.

This model is the one used to participate in CASP13 in 2018.

Input Features

For this network, the following features are generated from the sequence as the input, and each sequence has the real distance map as the ground truth.

1. Features for each residue, the features include the Position-Specific Scoring Matrix (PSSM), secondary structure prediction (3 states), solvent accessibility prediction (1 state), and the physicochemical features with dimension of 5. Therefore, for each residue, the feature vector length is 29. We call this set of features 1D feature.
2. Features for each residue pair, includes three evolutionary coupling scores from EVFold, PSICOV, and CCMPred respectively, and the mutual information, the normalized mutual information, and the contact potential, the dimension of those features are all 1 for each. Therefore, for each residue pair, the feature vector length is 6 and for a protein sequence with length L , the feature dimension is $L \times L \times 6$. We call this set of features 2D feature.

Network Structure

Unlike the previous network, instead of concatenating all features together in advance, the 1D and 2D features are processed separately. To be more specific, the 1D features are processed at the early stage in the network, then the results will be combined with the 2D features in the later stage in the network. The following Figure 4.8 shows a flowchart of the features used in this network.

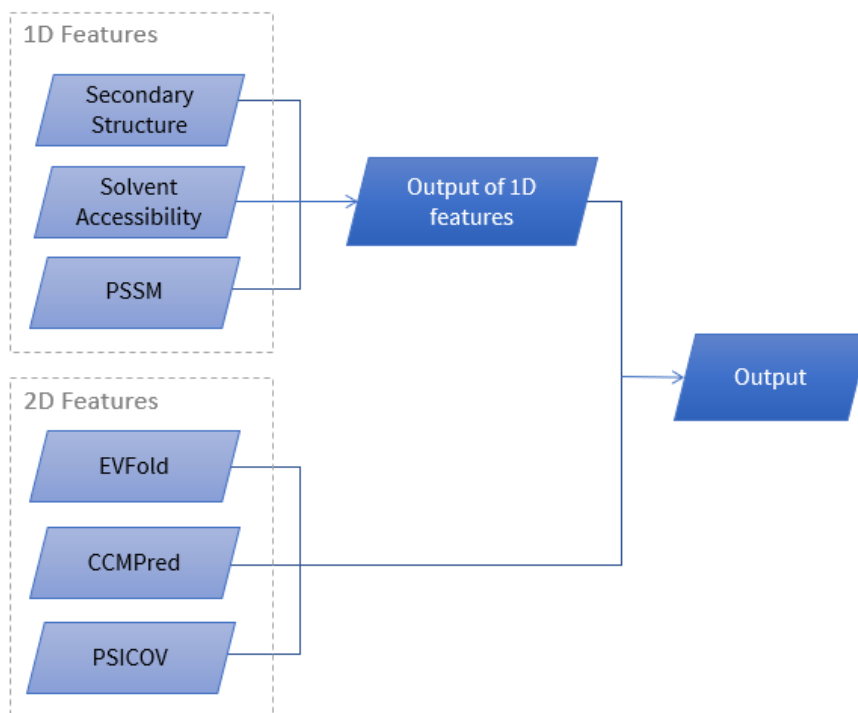


Figure 4.8 The features flowchart in network.

The following Figure 4.9 shows the network structure. The 1D features will be processed by a branch with 3 residual blocks. The filter size is 9×9 and the number of feature map is 32. The output size of this branch is $L \times 32$, in order to combine it with 2D features, a residue-wise concatenation like outer product is done. Assume R_i is the i th residue in the sequence, so the R_i has a length-32 feature vector. After the outer layer, the output size is $L \times L \times 64$, in which each residue pair R_{ij} has a length-64 feature vector from the concatenation of R_i and R_j . In the second 2D residual block branch, there are 6 residual blocks with kernel size 5×5 and the output feature map size is 128. At the last stage of the network, there are three 1×1 convolutional layers which are doing a depth-wise fully connected operation. The number of feature map is $128 \rightarrow 64 \rightarrow 32 \rightarrow 2$. The

final activation function is SoftMax and the number of categories is 2. In this way, each residue pair will be classified into one of two categories, i.e. in contact or not in contact.

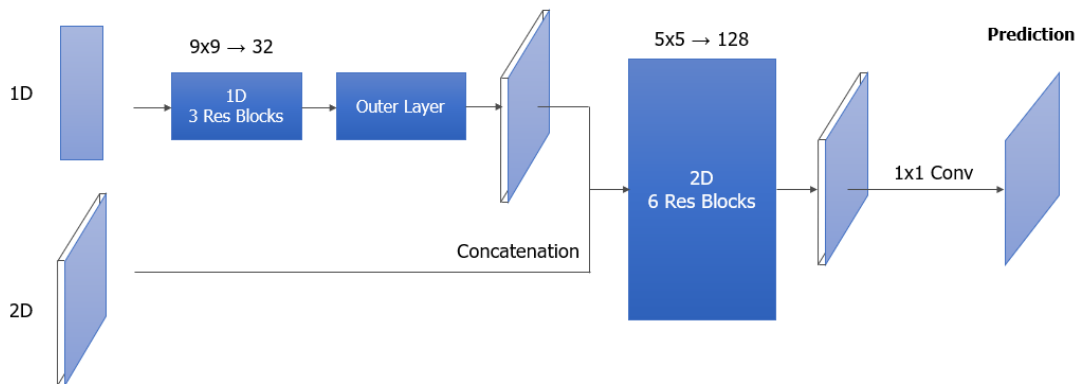


Figure 4.9 Network structure to predict binary contact directly.

The training set for this version of network is dataset version 1 for participation in CASP13, and this model is called the baseline. The baseline model has been kept improving after CASP13 with the following major updates:

1. Dataset version 2 is used to train the model. It is a larger dataset with 5250 training samples, while in dataset version 1 there are only 3799 training samples. This gives us around 5% improvements on long range top L/5 prediction precision on the testing set.
2. For the multiple sequence alignment searching tool HHBlits, we use a newer official released database. This gives us improvements on medium and short ranges prediction precision.

The loss function used is weighted cross entropy. The weights for class 0 and 1 are 0.1 and 0.9 respectively because a contact map is imbalanced and the amount of 0s is larger than amount of 1s. The optimizer is Adam and the learning rate is $1e-4$. The batch size is 1. i.e. for each iteration only one training sample is used to calculate the gradients and

update the weights in the network. A learning rate scheduler is also used in the training.

The initial training epoch is set to 15, the learning rate changes based on the epoch:

$$\text{Learning rate} = \begin{cases} 1e - 6, & 75\% \text{ of total epochs} \\ 1e - 5, & 50\% \text{ of total epochs} \\ 1e - 4, & \text{otherwise} \end{cases}$$

The testing performance will be discussed in the next Evaluation Results section.

4.4.3 Two-Stage Multi-branch Network

The distance map prediction and binary contact map prediction have been done separately. The distance map prediction is informative but not very accurate. The binary contact map prediction may lose the distance distribution information since the final classes are only 0 and 1. Based on these intuitions, a two-stage multi-branch network to combine the distance map and binary contact map prediction has been designed and developed.

In the two-stage multi-branch network, there are two stages. In the first stage, multiple networks predict the distance map for short, medium, and long range separately. Those predicted distance maps are then combined with our initial feature set in the second stage to predict the binary contact map. The distances are used as the intermediate results for contact prediction in the whole process.

The following figure shows the overall structure of the two-stages multi-branch network.

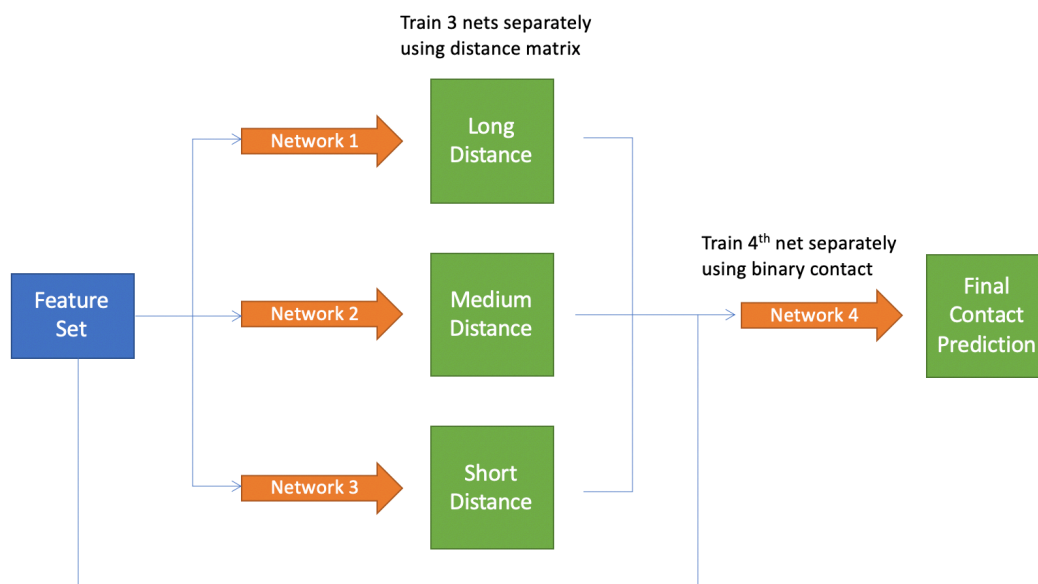


Figure 4.10 Network structure of the two-stages multi-branch network.

Input Features

Three more features are added in this version of network. In addition to all the features mentioned in the last Predicting Contact Map Directly section, the metaPSICOV output, the Hidden Markov Model (HHM) profile and Shape String are added. The HHM profile and Shape String are for each residue and they have the dimension of 30 and 8 respectively. The metaPSICOV output is for each residue pair and the dimension is $L \times L \times 1$. The final size for the 1D features is $L \times 67$ and for the 2D features is $L \times L \times 7$.

Stage 1: Distance Map Prediction

The basic structure of this network is like previous networks. There are 1D and 2D features, 1D features will be processed first, but instead of combining the output with 2D features directly, the 2D features will be processed before the combination. The following Figure 4.11 shows the structure of this network.

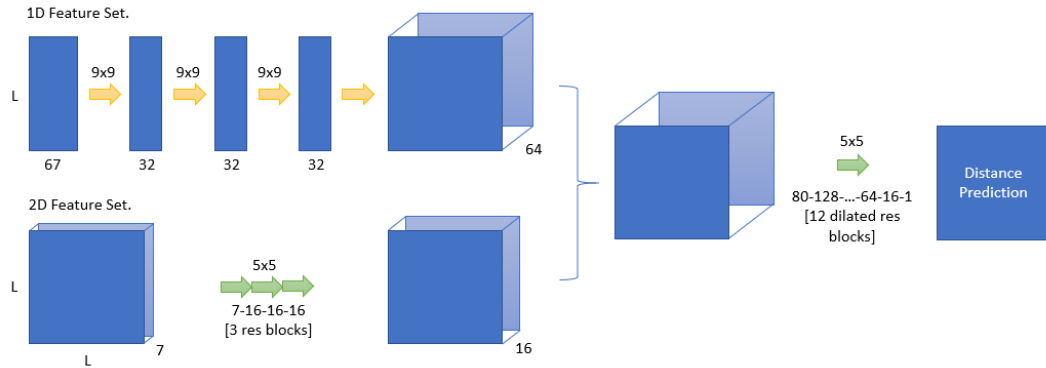


Figure 4.11 Network structure for distance prediction in the first stage in the two-stages multi-branch network.

In this network, the 1D feature set will go through a branch with three residual blocks, the number of the feature map is 32 for all convolution layers with kernel size 9×9 . The output size of this branch is $L \times 32$ and then after the residue-wise concatenation the feature size is $L \times L \times 64$. The 2D feature set will go through another branch with three residual blocks, the number of the feature map is 16 for all convolution layers with kernel size 5×5 . The output size of this branch is $L \times L \times 16$. After concatenate the output of those two branches, the feature will go through the main network with 12 dilated residual blocks (Yu & Koltun, 2015), and the number of feature map is 128 for all convolution layers with kernel size 5×5 except the last two 1×1 convolution layers, which are doing a fully connected operation for each residue pair and reducing the dimension from 128 to 64, and further to 1 at last. The dilation in the dilated residual blocks is 4. The activation function in the last layer is ReLU to get positive real values as the distance map prediction.

The dilated residual network is the traditional residual block with dilated convolution layers. The dilated convolution (Yu & Koltun, 2015) is also known as à trous algorithm (Fowler, 2005). It can greatly increase the size of receptive field in the convolutional layers.

The basic idea is to adding spaces between points in the kernel. The following Figure 4.12 shows the kernel points and the corresponding receptive fields in dilated kernel. From left to right, the dilation is 0, 1, and 3 respectively.

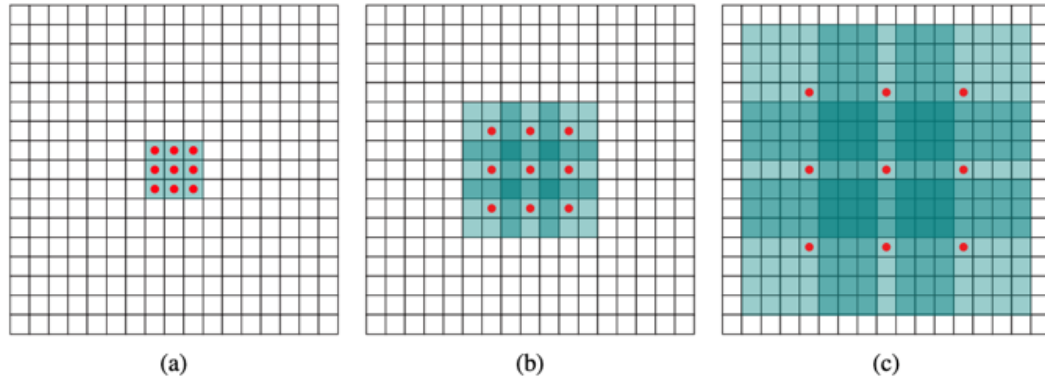


Figure 4.12 Dilated kernels and the receptive fields.

In this network, different dilation values have been tried and it is found that the best performance with dilation set to 4. In order to keep the input and output of a convolution layer the same size, the stride size is 1 for all layers and the padding size is calculated as in the following equation:

$$p = \frac{d \times (k - 1)}{2}$$

where the d is the dilation size, k is the kernel size.

Three networks with the same network structure will be trained for short, medium, and long-range prediction respectively. The intuition behind this is to let each network focus on each range to utilize the feature information.

Stage 2: Binary Contact Map Prediction

The output from stage 1 will be combined with the original feature set as the input to the network in stage 2. There are three distance map predictions in stage 1 so the additional

features to stage 2 have the dimension $L \times L \times 3$. The network structure used in stage 2 is the same as that in section Predicting Contact Map Directly except that the input size is different.

4.5 Evaluation Results

There are two datasets used to evaluate the performance. The first dataset is the CASP13 targets. The training set is dated before the start date of CASP13 so there are no native structures, i.e. the ground truth, contained in our training set. However, not all CASP13 targets have native structures, only 19 out of all targets are used because CASP has officially released the native structures for those 19 targets. The second dataset is a set of sequences with PDB structures released between 2019-04-01 and 2019-04-20. It contains 50 sequences and this dataset is called NewPDB50_Testing in this dissertation.

CASP13 Dataset

The following table shows the target list used for this evaluation.

Table 4.4 Target list used for contact map evaluation.

T0950, T0951, T0953s1, T0953s2, T0954, T0955, T0957s1, T0957s2, T0958, T0960, T0963, T0966, T0968s1, T0968s2, T1003, T1005, T1008, T1011, T1016

In our testing, domains are not used in those targets. It is slightly different from CASP official evaluation because the domain definition was not available when we started our early evaluation. The whole sequence of each target will be used as input. The performance of all other groups is evaluated by us because their predictions are publicly available to download.

CASP13 Dataset Results

Because the CASP is using the long-range top L/5 predictions to rank the group by default, only the results for this specific metric are showed. The long-range contacts are also more important than shorter range because they are hard to predict, and they decide the global structure of a protein.

The progress can be divided into several milestones:

1. Predict distance map directly.
2. The baseline model we used to participate in CASP13 to predict contact map directly.
3. Using a larger training set.
4. Cleaning and filtering the training set.
5. Using a newer HHBlits database to generate MSA.
6. Two-stage multi-branch network with four networks in stage 1 (four networks are for short-, medium-, long-, and full-range distance predictions).
7. Two-stage multi-branch network with three networks in stage 1 (the same as milestone 6 but with the full-range distance prediction network removed).

The following Table 4.5 shows the long-range top L/5 predictions precision for each of the milestone and the corresponding rank in CASP13 using the precision value.

Table 4.5 The long-range top L/5 precision for each milestone and the corresponding ranks.

Milestone	Long-range top L/5	Rank
1	38.38%	38
2	49.25%	28
3	49.51%	28
4	53.73%	19

5	55.68%	17
6	63.81%	14
7	66.62%	8

From the testing result, our current two-stages multi-branch model gets the best performance and it has 35.27% increase of the baseline model, i.e. the model we used to participate in CASP13. The following Figure 4.13 shows the plots of our milestone results and the corresponding ranks.

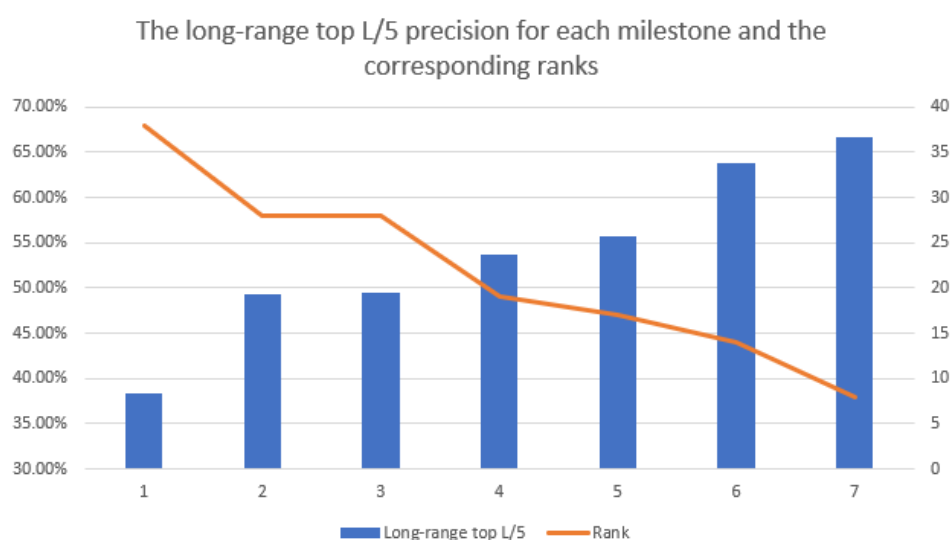


Figure 4.13 The graph for long-range top L/5 precision for each milestone and the corresponding ranks.

In addition, other four widely used downloadable tools are chosen to compare the performance without proposed methods. All tools are downloaded and installed on our server to test. Online servers are excluded since we don't know if their training set contains natives of our testing set. The four tools are:

1. FreeContact
2. PSICOV
3. CCMPred

4. metaPSICOV2

The following table shows the performance of the comparison.

Table 4.6 Contact prediction precision comparison with other tools using CASP13 targets.

Tools	Long			Medium			Short		
	L/2	L/5	L/10	L/2	L/5	L/10	L/2	L/5	L/10
FreeContact	0.071	0.092	0.147	0.077	0.105	0.165	0.085	0.093	0.124
PSICOV	0.224	0.279	0.365	0.155	0.233	0.365	0.155	0.225	0.365
CCMPred	0.227	0.277	0.395	0.174	0.266	0.411	0.175	0.225	0.395
metaPSICOV	0.337	0.450	0.526	0.404	0.562	0.747	0.409	0.576	0.784
MUFOLD Contact D	0.282	0.408	0.482	0.295	0.384	0.482	0.204	0.258	0.335
MUFOLD Contact C	0.388	0.492	0.574	0.411	0.578	0.763	0.382	0.536	0.742
MUFOLD Contact	0.542	0.666	0.726	0.458	0.615	0.663	0.443	0.675	0.752

The following figure shows the visualization of the results:

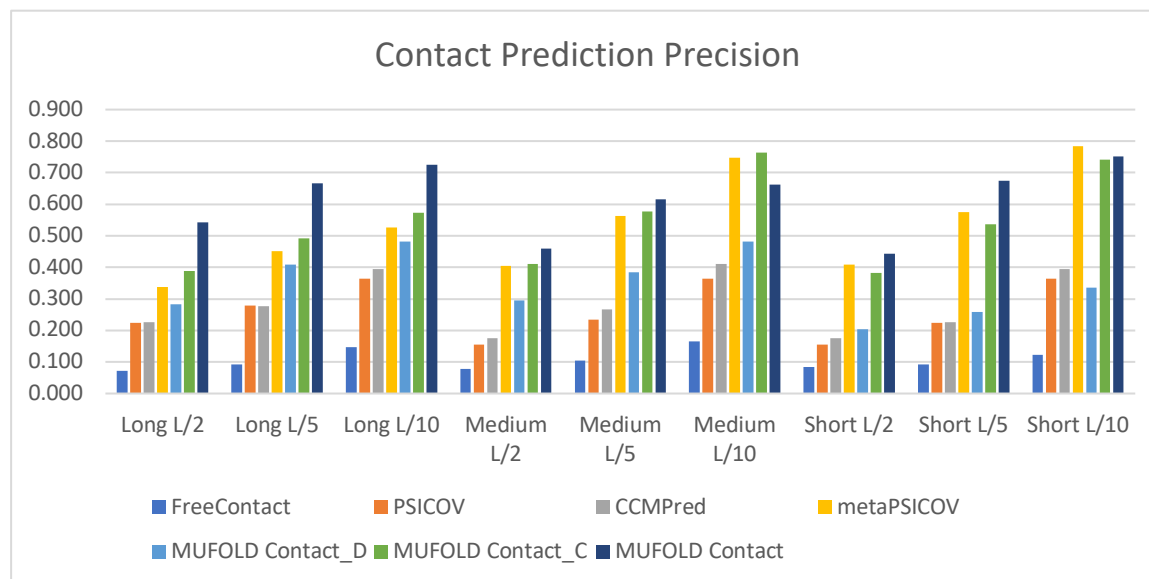


Figure 4.14 Chart of contact prediction precision.

NewPDB50_Testing Dataset

In order to have a fairer comparison of each predictor, we decided to make a new testing dataset with the recently released PDBs to avoid homologies. We selected all PDBs that

are released between 2019-04-01 and 2019-04-20. Of those structures, only those with length between 100 and 400 and maximum sequence identity of 30% were considered. CD-Hit was further applied to remove duplicate chains. Finally, 50 sequences were selected randomly as our evaluation dataset.

NewPDB50_Testing Dataset Results

For this dataset, other five widely used tools are chosen to compare the performance. All tools except RaptorX-Contact are downloaded and installed on our server to test. The five tools are:

1. EVfold
2. CCMpred
3. metaPSICOV2
4. ResPRE (Y. Li et al., 2019)
5. RaptorX-Contact (online server)

The following table shows the performance of the comparison.

Table 4.7 Contact prediction precision comparison with other tools using NewPDB50_Testing targets.

Method	Long			Medium			Short		
	L/10	L/5	L/2	L/10	L/5	L/2	L/10	L/5	L/2
EVfold	0.111	0.153	0.212	0.083	0.106	0.12	0.077	0.074	0.079
CCMpred	0.412	0.589	0.661	0.205	0.353	0.496	0.173	0.293	0.432
MetaPSICOV	0.591	0.733	0.783	0.407	0.606	0.733	0.379	0.591	0.728
ResPRE	0.779	0.868	0.89	0.522	0.763	0.852	0.376	0.651	0.787
RaptorX-Contact	0.799	0.875	0.899	0.547	0.766	0.849	0.465	0.721	0.815
MUFold-Contact	0.756	0.861	0.906	0.506	0.747	0.843	0.461	0.723	0.828

From the above table, MUFold-Contact performed significantly better than EVfold, CCMpred, and MetaPSICOV, and it is better than ResPRE on short-range cases, slightly worse on medium-range cases, and is comparable on long-range cases. Compared to RaptorX-Contact, MUFold-Contact is slightly better on short-range L/10 and L/5 cases and slightly worse on other cases. Overall, MUFold-Contact is comparable with the two deep learning methods ResPRE and RaptorX-Contact.

4.6 Application: Contact Guided Modeling

MUFOLD_Contact is used in our in house comprehensive protein structure prediction platform, MUFOLD, to assist the 3D structure modeling process. For introduction of MUFOLD please refer to Chapter 6 in this dissertation. The traditional template-based protein structure prediction methods rely on the quality of templates. If there is no good template found in the database, the quality of the predicted protein structure could be very poor. This is where the MUFOLD_Contact can help since the contact map provides the global topology of the sequence and this topology information can be used as constraints to refine the predicted protein structure.

There are some related works that use this idea in their protein structure prediction application. Crystallography & NMR System (CNS) (Brunger, 2007; Brunger et al., 1998) is a tool that uses distance geometry simulated annealing protocol to import the residue pair constraints in modeling process. Confold2 (Adhikari & Cheng, 2018) is based on CNS. C-I-TASSER (Zheng et al., 2019) is another contact guided protein structure prediction tool.

In order to demonstrate the usefulness of MUFOLD_Contact results and also to improve the performance of MUFOLD, we implemented an iterative contact map guided modeling based on Rosetta (Rohl, Strauss, Misura, & Baker, 2004). Our method is very preliminary but shows some good sign of the performance. The method architecture is shown in the following Figure 4.15.

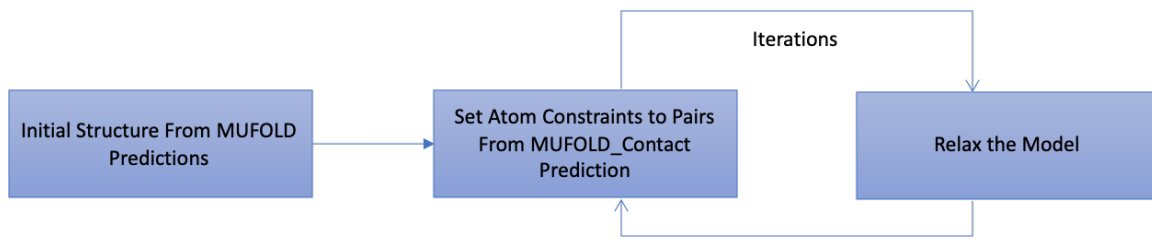


Figure 4.15 Flowchart of contact map guided modeling.

Firstly, the structure prediction from MUFOLD is used as the initial structure in this process. Then the contact map prediction will be used as atom constraints to a set of residue pairs in the sequence. The set of residue pairs in each iteration is decided by a sequence distance threshold of 10, which means in the first iteration, only residue pairs within sequence distance of 10 will be given constraints, and in the second iteration, only residue pairs within sequence distance of 20 will be given constraints, etc. Experiments showed this iteration process gave better results because it is a continues refinement process from the local structure to global structure rather than refining on global structure directly. Rosetta is used in the relaxation process.

We have performed experiments on some of CASP13 targets and the first results we found is this method only works on hard target, which is the sequence with no good templates. This is consistent with our previous assumptions. Each target was run five times to get five separate results. The following Table 4.8 shows the preliminary results.

Table 4.8 CASP13 results on contact map guided modeling.

Target	Initial Model	Best	Worst	Avg
T0951	0.9164	0.7961	0.7481	0.7664
T1016	0.7698	0.7661	0.7141	0.7404
T0966	0.5508	0.1153	0.0600	0.0804
T0958	0.3929	0.4156	0.2825	0.3299
T0957s2	0.3703	0.4430	0.3623	0.4000
T0968s2	0.2217	0.3370	0.1848	0.2313
T0953s1	0.2118	0.3125	0.2396	0.2694
T0968s1	0.2034	0.3008	0.2436	0.2733
T0957s1	0.1728	0.1836	0.1420	0.1645
T0954	0.1484	0.1155	0.0760	0.0917
T0950	0.1440	0.1988	0.0885	0.1259
T0953s2	0.0796	0.1139	0.0927	0.1026
T0963	0.0522	0.0962	0.0687	0.0846
T0960	0.0394	0.0842	0.0635	0.0726
Avg (Excluded first three targets)	0.1851	0.2365	0.1677	0.1950

The corresponding results are also showed in the following Figure 4.16.

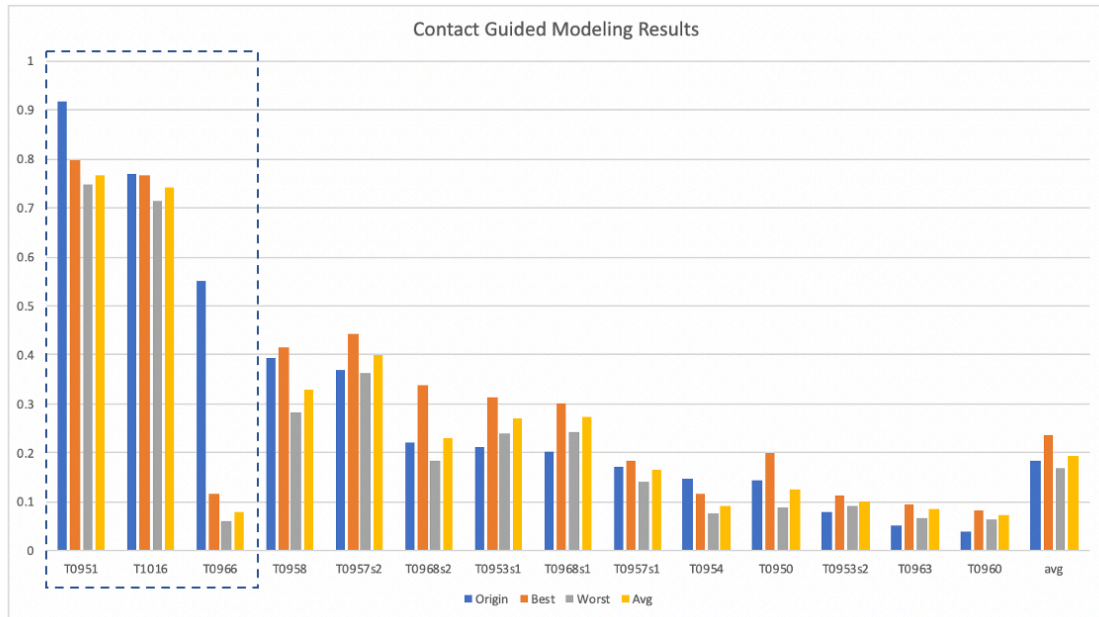


Figure 4.16 CASP13 results on contact map guided modeling.

In Table 4.8 and Figure 4.16, targets are sorted by the GDT-TS of the initial model. The “Best”, “Worst”, and “Avg” columns mean the best, worst, and average GDT-TS in the five runs. From the results we have preliminary results that our method works better on hard targets than easy targets. Without considering the first three targets in the table, our contact map guided modeling can give 5.35% improvement on average and 27.77% improvement if ideally the best result in five runs could be chosen. Potential value of our MUFOLD_Contact results are proved to be helpful to assist the structure prediction.

4.7 Summary

In this chapter the progress of our deep learning based contact map prediction method has been shown. The idea of Fully Convolutional Network (FCN) is applied to make the network be able to accept various length of protein and the idea of Residual Network is to make it deeper. Different ways have been explored to get the contact map prediction from the sequence. The first way is to predict distance map directly from the sequence and pairwise features. It suffers from the regression precision and the distance range limits. The second way is to predict contact map directly. Each residue pair is classified into two classes. It suffers from the loss of information from the features set. The third way is a two-stages multi-branch network that is to combine the previous two ideas and use distance map predictions as an additional feature to predict contact map. There are two stages, the first stage for distance map predictions using multiple networks for different range of residue pairs, the second stage for contact map prediction. In addition, more features and dilated convolution are introduced to get a better performance. The third way is proved to be the best current model. It is also demonstrated that our preliminary results showed using

our predicted contact maps as constraints in the structure prediction for targets without good templates can give better results than single template-based modeling.

CHAPTER 5. TPCREF: PREDICTED CONTACT MAP REFINEMENT

5.1 Motivations

Most current state-of-the-art contact prediction methods are not yet capable of correctly predicting all contacts for a given amino acid sequence. In addition, most of existing contact prediction methods are based on co-evolutional information, which are extracted from sequences only. They lack using the information from templates, while it is well known that templates carry the most important structural information in the tertiary structure modeling. Therefore, if we can find a method to utilize those templates' structural information there is a chance that we can refine and improve the quality of predicted contact map of the target sequence. Additionally, since existing contact predictors have different methods and use different features, it is more important to make the refinement independent of the existing predictors so that we can put it after the contact map prediction as a post process.

Since the definition of templates is those have similar structures to the target sequence. Each of the templates has a similarity score defined by the searching tool. The similar idea happens in the recommendation system, especially the user-based collaborative filtering. Before we present our method, let's take a brief look at the basic process of user-based collaborative filtering.

As showed in the following figure, there are multiple users and items. Each user has a score for each item. In order to predict the score of U_1 for $Item_1$, we perform the following process:

	<i>Item₁</i>	<i>Item₂</i>	...	<i>Item_m</i>
<i>U₁</i>	...	?
<i>U₂</i>	...	S_{22}
...
<i>U_n</i>	...	S_{n2}

Figure 5.1 An table to illustrate the idea of user-based collaborative filtering.

1. Find a set of users that are most similar to U_1 and also have rated a score for $Item_1$.
The similarity score between U_1 and U_n is defined as Sim_{1n} ;
2. To predict the score S_{12} , a weighted average of other scores for this item from other similar users is calculated. This weighted average score will be used as the predicted score of U_1 for $Item_1$.

Our template-based predicted contact map refinement method is inspired by the idea from user-based collaborative filtering. For a predicted contact map, we can assume there is a quality score for each predicted residue pair. In order to get this quality score matrix, we can find a set of similar predicted contact maps, of which each residue pair already has a quality score of the contact map predictor. If the quality scores of the predicted contact map is U_1 , then the quality scores of those similar predicted contact maps are other similar users.

Under the inspiration of user-based collaborative filtering, TPCref (refinement by template prediction correction), a general method for refinement which can be applied to any contact prediction method, is proposed in this research. It refines a method's prediction for a given target sequence by utilizing knowledge of misclassified contacts from

predictions for template sequences. In our background evaluation, comparable methods for refining contact map predictions were not found, and we are the first to utilize idea from collaborative filtering to help the refinement process.

5.2 Problem Formulation

The predicted contact map refinement problem is addressed as follows. Give a protein sequence S and an existing contact map predictor P , we can have a contact map prediction C . We don't know how P works but P should be able to take the sequence S as input and predict C . Then TPCref takes the contact prediction C as input, and also has the access to call predictor P during the refinement. The final refined contact map prediction C_R will be outputted as the final contact map prediction of sequence S .

5.3 Overall Architecture

TPCref consists of contact prediction using the contact prediction method of interest, template selection, and assembly of a target contact-map filter, depicted as a flowchart in the following Figure 5.2.

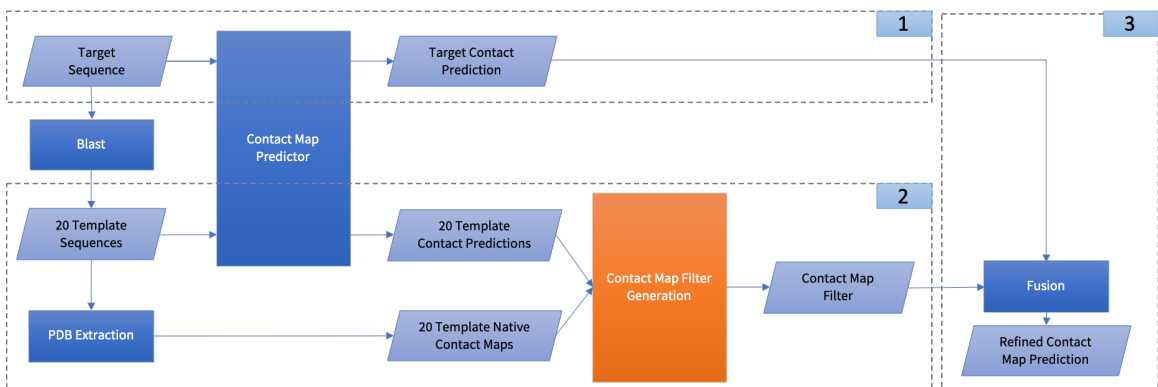


Figure 5.2 The flowchart of TPCref

As we can see from the flowchart, there are three major components. First, given a target sequence and a contact map predictor, we can get an initial target contact map prediction. Then in the second part, there are 20 template sequences selected from the Blast results. Those 20 templates have native structures in the PDB databank so we can extract the native contact maps of them. In the meantime, those 20 template sequences will also get their predicted contact maps by the given contact map predictor. Then a contact map filter is created by using the information from 20 templates' native contact map and predictions. Finally, the contact map filter will be fused to the original target contact map prediction to get the refined contact map prediction.

5.4 Refinement Process

Given a target sequence and an existing contact map predictor, TPCref performs the following processes to get the final refined contact map prediction.

5.4.1 Template Selection

Templates of a target protein sequence are generated using an iterative, three-stage Blast search process. The first and second stages perform a Blast search on the non-redundant (NR) sequence database, with the results of the first stage acting as the starting point of the second. The third stage utilizes the results of the second stage to search in a database containing sequences with known structures in the PDB database. The top 500 or fewer templates returned from the Blast are clustered by their full-sequence identity to yield about 20 final templates. The lowest E-value members of resulting clusters are used.

5.4.2 Contact Map Prediction Using an Existing Method

After 20 most similar templates are selected, we need to do contact map prediction as well as real contact map extraction for each of them. We tested an extensive set of existing contact prediction methods, including EVfold, CCMpred, MetaPSICOV, ResPRE, and MUFold-Contact. These methods were selected because their software could be downloaded to perform the extensive computation and experiments. Because RaptorX-Contact method is only offered through an online server, we could not apply TPCref to it.

5.4.3 Template Contact Map Filters Generation

The contact prediction by an existing method for each of the templates will be used to form a template contact map filter. The filter is of the same size as the contact map and the generation process is as follows.

For each template T_i , we have the contact map prediction P_i from previous step, and the corresponding native contact map N_i which is extracted from the PDB structure. To build a filter F_i for this template, the following equations are applied.

For short and medium range residue pairs $R(m, n)$:

$$F_i(m, n) = P_i \times 2 \times (N_i(m, n) - 0.5)$$

For long range residue pairs $R(m, n)$:

$$F_i(m, n) = P_i \times N_i(m, n)$$

By applying the above equations, for each short and medium residue pair position in the filter, if this position in the template's native contact map is 1 (in contact), then set the value to the predicted contact probability by the existing method. If this position in the template's native contact map is 0 (not in contact), then set the value to -1 times the

predicted contact probability by the existing method (thus penalizing false positives). For each long residue pair position in the filter, the only difference is when this position in the template’s native contact map is 0 (not in contact), set the value to 0 instead of -1 times the predicted contact probability.

The predicted probabilities were used to construct the template filters rather than the true contacts in order to exploit behaviors which are characteristic of the method being refined.

5.4.4 Predicted Contact Map Refinement

Now for each template T_i there is a filter F_i . For each residue pair (m, n) , the corresponding value F_{i-mn} in the filter can be treated as a quality score of the predictor for that specific residue pair. It is like a rating system that templates act as other similar users, and template filter acts as the ratings of those users. By taking the “quality recommendations” from other templates we can do a prediction of the quality scores CF_{mn} , which is the contact filter of the original contact map. This comparison of this idea and traditional user-based collaborative filtering is showed in the following Figure 5.3.

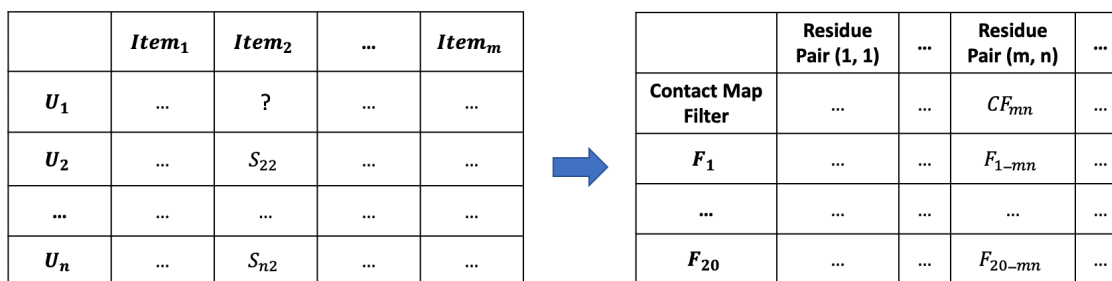


Figure 5.3 Comparison between the TPCref contact map filter generation and the traditional user-based collaborative filtering.

In traditional collaborative filtering, the prediction is a weighted average of other users' ratings and the weight is the similarity between the user to be predicted and other similar users. In our problem, we set the weight to the percent coverage of each template's respective sequence on the target sequence as returned from the Blast search process. Then each residue pair position in the target contact map filter is set to the weighted average of the aligned position's value from all individual template filters covering that position. Finally, each position in the target filter was scaled by the natural log of the number of covering templates, in order to reinforce positions with more contributions from template filters.

The final contact prediction of the target protein is the sum of the initial contact map prediction by an existing method and the target contact map filter, while negative values were set to 0 and all values were normalized to be in the range of 0 to 1.

5.5 Evaluation Results

5.5.1 Dataset

To compare the performance of using TPCref and without using TPCref, we use the NewPDB50_Testing dataset to test. The details of NewPDB50_Testing dataset is introduced in Chapter 4.5.

5.5.2 Results

TPCref was applied to the predictions of five methods (EVfold, CCMpred, MetaPSICOV, ResPRE, MUFold-Contact), utilizing the same set of templates. The prediction accuracy of the refined contact predictions is reported in the following Table 5.1. Compared to the results in Table 4.7 that are without TPCref, we can see that TPCref

significantly improved on the baseline accuracy in nearly every case. The improvements of accuracy compared to without TPCref is showed in the following Figure 5.4. As shown in the figure, all values are positive means TPCref can improve the accuracy of predicted contact map on every metric.

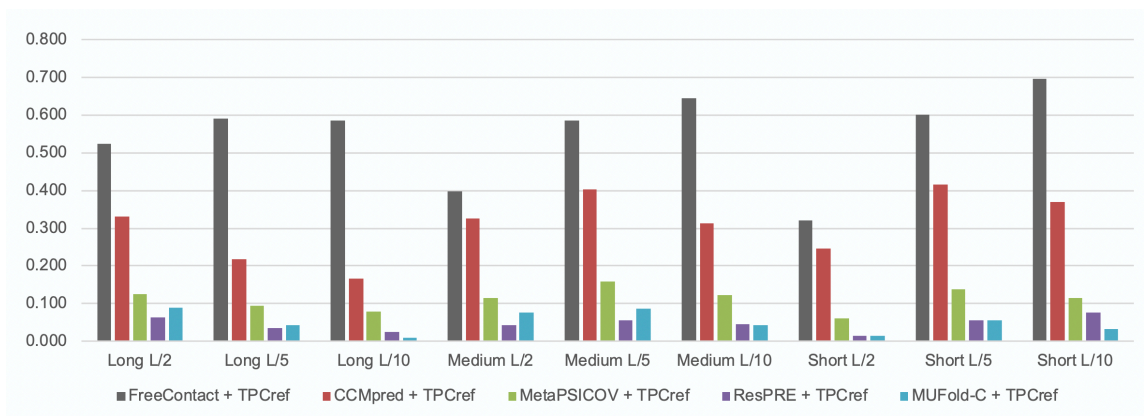


Figure 5.4 Improvements after applying TPCref

Notably, for the long L/5 accuracy results, TPCref achieved improvement on all existing methods. In this result, MUFold-Contact+TPCref obtained the highest accuracy, outperforming all existing methods, including RaptorX-Contact.

Table 5.1 TPCref refined contact prediction precision comparison with other tools using NewPDB50_Testing targets.

Method	Long			Medium			Short		
	L/10	L/5	L/2	L/10	L/5	L/2	L/10	L/5	L/2
EVfold + TPCref	0.634	0.744	0.799	0.480	0.691	0.766	0.397	0.674	0.775
CCMpred + TPCref	0.743	0.808	0.827	0.530	0.755	0.808	0.420	0.709	0.801
MetaPSICOV + TPCref	0.716	0.827	0.863	0.522	0.766	0.857	0.441	0.728	0.843
ResPRE + TPCref	0.842	0.903	0.915	0.566	0.820	0.898	0.391	0.707	0.863
MUFold-C + TPCref	0.845	0.904	0.915	0.582	0.834	0.887	0.477	0.778	0.861

Among all the metrics, long L/5 is the most important one and is used by CASP to rank groups by default. The long L/5 accuracy results for each method from Table 4.7 and the corresponding refined results from Table 5.1 are summarized in the following Figure 5.5. All methods showed improvement in long L/5 accuracy when TPCref was applied, though the state-of-the-art methods had less room to improve.

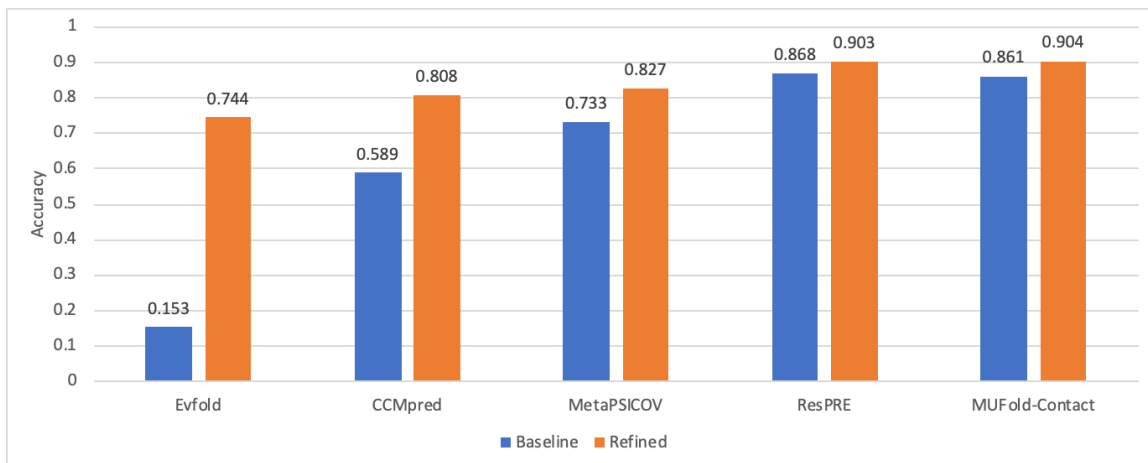


Figure 5.5 Long L/5 results comparison of NewPDB50_Testing targets.

As described in Chapter 2.4.2, diversity is another metric to evaluate the performance of contact map refinement. The diversity of the refined contact predictions is reported in following Table 5.2 and 5.3. In every case, the diversity of contact-map predictions increased with TPCref. Refinement of the ResPRE and MUFold-Contact contact-map predictions achieved only minor increases in diversity, similar to the results for prediction accuracy. This is notable, as it has been hypothesized that template-based techniques may contribute to a lack of diversity in predicted contact maps (Y. Li et al., 2019).

Table 5.2 Shannon entropy of contact map predictions by MUFOLD-Contact and existing methods.

Method	All	Long
EVfold	0.044	0.207
CCMpred	0.507	0.675

MetaPSICOV	0.928	0.799
ResPRE	1.146	1.040
RaptorX-Contact	1.080	1.044
MUFold-Contact	1.110	0.981

Table 5.3 Shannon entropy of contact map predictions by TPCref and existing methods.

Method	All	Long
EVfold + TPCref	0.630	0.931
CCMpred + TPCref	1.062	1.079
MetaPSICOV + TPCref	1.082	0.944
ResPRE + TPCref	1.198	1.113
MUFold-C + TPCref	1.169	1.092

5.6 Summary

In this chapter we proposed TPCref, a method for the refinement of contact predictions. Starting with a target sequence and a prediction method, this method utilizes contact predictions made on a set of clustered templates to produce a refined target contact-map prediction. For a refinement method to be effective, it should improve or at least maintain performance compared to the baseline prediction. This method was applied to the contact predictions from five methods (EVfold, CCMpred, MetaPSICOV, ResPRE, and MUFold-Contact) on a dataset of 50 sequences with recently released PDB structures, which is called NewPDB50_Testing. This selection of methods used for evaluation includes a diverse set of approaches, with performances spanning a wide range. TPCref showed increased performance in both accuracy and diversity of contact predictions in nearly every metric considered. This performance suggests that TPCref is suitable for application to contact predictions from a wide variety of methods.

CHAPTER 6. MUFOLD PLATFORM DEVELOPMENT

In this chapter, the comprehensive protein structure prediction platform, MUFOLD, and its architecture, functionalities, methods and algorithms, are introduced.

6.1 Introduction

MUFOLD is a solution for protein structure prediction (Jingfen Zhang et al., 2010). It provides an end-to-end solution for various structure prediction tasks. There are many similar platforms like Rosetta (Rohl et al., 2004), QUARK (Xu & Zhang, 2012), MULTICOM (J. Li et al., 2014), I-TASSER (J. Yang & Zhang, 2015), etc. MUFOLD can help experimental biologist to have a better understanding of protein structures and accelerate the process of experiments.

In general, MUFOLD is one of the template-based modeling methods (Fiser, 2010). The basic idea is for a new sequence target, MUFOLD tries to search and find a set of similar sequences with known structures according some similarity measurement. Those similar structures are called templates. Some targets are easy to model if a very similar known structure can be found in database. Some targets are hard to model if there are no similar known structures.

6.2 Contributions

1. The original MUFOLD source codes that was written in C a long time ago have been factored. The idea of Objected Oriented Programming has been introduced and all codes have been rewritten using C++ and Python.

2. Different modules have been decoupled by their functions. Each module can talk with each efficiently other using our own well-defined protocols. This design also gives us the ability to add or change functions in each module much easier so that it is possible to have a fast testing iteration if there are new ideas or tools that we would like to use.
3. More tools have been integrated to provide more functionalities. New tools that have been added include a new deep learning based loop modeling method, a new deep learning based secondary structure and supersecondary structure predictor, a new deep learning based contact predictor, and a new quality assessment method, etc.

6.3 System Architecture Overview

The whole pipeline of MUFOLD structure prediction can be divided into five modules as shown in the following Figure 6.1. Each module can run individually or can be integrated to other tools using our pre-defined API protocol. Each module will be gone over in details in the following sections.

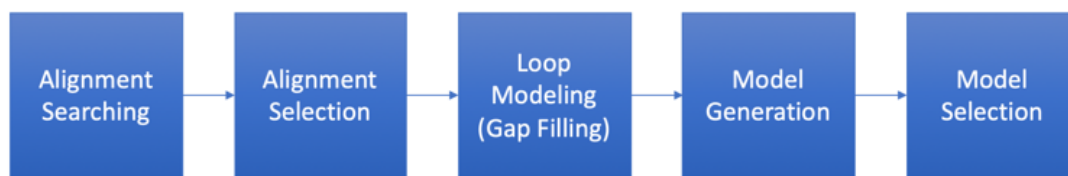


Figure 6.1 System architecture of MUFOLD pipeline.

6.4 Database Preparation

Multiple alignment searching tools are using in MUFOLD, i.e. Blast (Altschul et al., 1997), HHSearch (Soding, 2005), and CNFSearch (Ma, Peng, Wang, & Xu, 2012). MUFOLD uses its own generated databases for Blast and HHSearch and uses the

CNFSearch's official released database for CNFSearch. Here we briefly describe the process of own database preparation and the details can be found in the MUFOLD_DB paper (Z. He et al., 2014).

The first step is to synchronize all the new released PDB files from PDB bank to local storage. It is important to keep a copy of all released PDB files for the convenience of data processing. Then a list of redundant PDB chains is got and all the information in PDB file is extracted to a separate file in our own format. The information what are extracted includes secondary structure, coordinates, Psi-phi angles, etc. Sequences are extracted from PDB file to a FASTA file as well. The Blast database will be built from those redundant sequences. Then, PSI-Blast is used on each sequence to get a multiple sequence alignment and a profile. An HHM database is generated from the MSA and profile for HHSearch.

6.5 Template Searching and Selection

Since MUFOLD is a template-based modeling method, the first step is to search and find similar templates and select the best templates for modeling.

For template searching, MUFOLD has integrated three alignment searching tools: Blast, HHSearch, and CNFSearch. Blast is a sequence-sequence searching tool, while HHSearch is a sequence-profile or profile-profile searching tool which considers the structural information during the searching process. CNFSearch is an algorithm based on Conditional Neural Fields.

Basically, the Blast contains three iterations. The first two iterations search on a non-redundant sequence database to generate a sequence profile. The last iteration will take this profile as input and search on our own generated PDB sequence database to get a final

alignment result. The HHSearch takes the profile from Blast's first two iterations as input and does a profile-profile search on our own generated HHM PDB database to get the alignment result. In addition, HHSearch is used twice with different searching mode: global search and local search. Therefore, two HHSearch alignments outputs are used. CNFSearch takes the query target sequence as input and searches on its own database to get the alignment result.

6.5.1 Template Selection

All the templates are divided into two types: the global (master) templates, and the local (fragment) templates. The master template is usually one template that is the best choice for the query sequence. The fragment templates will be used to create a candidate pool to fill in the missing areas or un-aligned part by the master template in the query target.

Each alignment searching tool has its own ranking algorithm and it is not comparable between each other's results. Therefore, a naïve consensus method is implemented to select templates from all three tools outputs. The intuition behind the consensus method is better templates have a higher probability to be more similar between each other (Fink, Kosecoff, Chassin, & Brook, 1984). For each template, the normalized TM-scores are calculated with all the other templates individually, then select the ones with higher scores. The normalized TM-scores mean the raw TM-score is normalized by the target length L (Sitao Wu & Zhang, 2007).

Master Alignment

There are 30 master alignment candidates and 10 from Blast result, 10 from HHSearch global search result, and 10 from CNFSearch result. For each tool, all the alignments are

ranked by their own ranking algorithm. To ensure the quality of master alignment, a coverage threshold is used to filter the master alignment. The default coverage threshold is 50% but in case of some very hard targets with only a few numbers of templates, the threshold value is gradually decreased until there are at least 10 templates available.

Then for each of the 30 master template candidates, the existing aligned C_α coordinates are extracted from the template's PDB file. The sequence ID for each C_α atom is re-numbered to the corresponding target sequence ID. This step will make sure when we calculate the consensus TM-score, the correct structures are compared.

A 30×30 consensus score matrix will be calculated for all 30 master template candidates. Each score in the matrix is the normalized TM-score of each pair of models. The average consensus score is calculated using the following equation and is used to select the top 10 master templates. Each master templates will lead to generated model at the end.

$$Consensus_Score_i = \frac{1}{29} \sum_{j=1}^{29} TMscore_{ij}$$

In the above equation, $TMscore_{ij}$ is the normalized TM-score of templates T_i and T_j .

Fragment Alignment

Fragment alignments are used to fill in the missing areas or un-aligned part by the master template in the query target. For example, if the master alignment only covers the 90% of the target sequence, then the rest of 10% sequence will be covered by further searching in the fragment alignment pool.

All fragment alignments are from Blast and HHSearch alignment results, excluding the selected master template. For each gap in the query target, several patches from the

fragment alignments pool will be searched and evaluated by some metrics (details in the later loop modeling section).

6.5.2 Template Representation

In MUFOLD, all templates are represented using distance matrix generated from its coordinates. The values in the distance matrix represent the real physical distance of each amino acid pair in the 3D structure. The advantages of using distance matrix are for the convenience of manipulating structures in 2D data rather than 3D data. At the end of modeling, we can use multidimensional scaling to convert distance matrix back to 3D structure (Jingfen Zhang et al., 2010).

6.6 Loop Modeling

Since we can't ensure all master alignments can cover the whole range of query target sequence, there are always missing areas or un-aligned part left. Those areas are called gaps. The gap has sequence data but doesn't have structure data. The method to fill in those gaps are called loop modeling. It is a very import step in MUFOLD since it ensures the completeness of a model, otherwise the model generated will be incomplete (Levefelt & Lundh, 2006).

Generally, all gaps are divided into two types: the gaps in the sequences, and on the terminal ends. For the gap in the sequence, it is usually not very large and have context on both gaps ends. For the gap on the sequence terminal ends, it could be very large and there is no context on one gap end.

6.6.1 Loop Modeling by Fragment Alignments Searching

In this method, only the gaps that are longer than three are considered. For the gaps that are smaller than three, the shortest path method described in the next section is used. For each gap in the query sequence that has no structural data, the gap ends are extended by three residues on both sides. For example, if the original gap length is 5, then the extended length is 11. The extended residues have structural data and will be used to select the best patches. In the following Figure 6.2, on the left side it shows a template sequence with a gap in it. The gap is then extended by three residues on both sides and the patch will need to be able to cover the whole length of the extended gap.

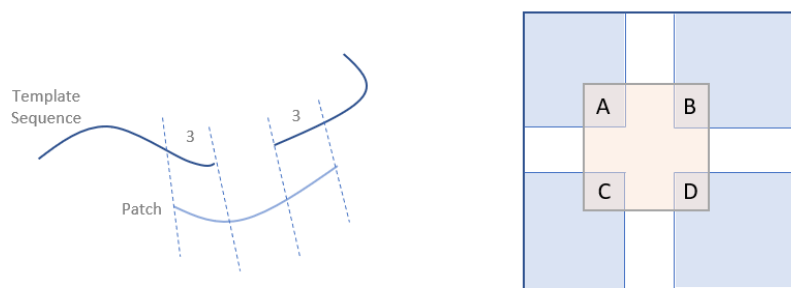


Figure 6.2 Illustration of overlap regions in distance matrix in loop modeling.

To find the patch candidates, the program first searches the all fragment alignments to find segments that can cover the whole extended gap. As shown in the right side in Figure 6.2, the blue square is the distance matrix of the template sequence. The gap will be two bands with no values as shown in the white area. The patch's distance matrix is shown in orange and the extended three residues will be the areas A, B, C, and D.

For the patch candidate's selection, it is done by a score called the root mean square distance (RMSD) of the overlap regions of the patch and the template sequence, i.e. the six

residues with three on each end. A smaller RMSD value indicates the overlap region between the patch and the template is more similar. Before calculating the RMSD, the six residues in the patch will be superimposed to the corresponding six residues in the template sequence. After the superimposing, the RMSD of the overlap region is calculated and used to sort the patch candidates. The patch candidate with the smallest RMSD will be selected for the gap.

The patching process will be based on distance matrix, as shown on the right side in Figure 6.2. The following Figure 6.3 shows an example of patching a gap in a distance matrix of a master template.

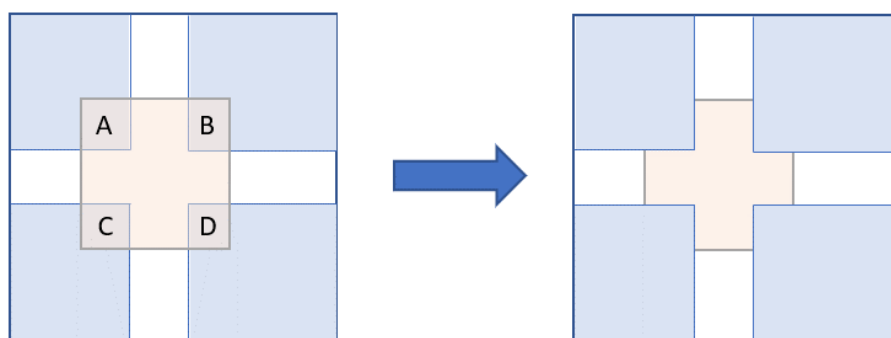


Figure 6.3 Illustration of loop modeling process by patching a gap in the distance matrix.

In the above Figure 6.3, the blue area in the square on the left side is the distance matrix of a master template with a gap in it, the orange area is the patch we select for this gap. The square on the right side is the distance matrix after the patch and the master template are merged. The blue area is fixed and only the gap area will be filled using the patch's distance matrix.

6.6.2 Loop Modeling by Shortest Path

For the gaps that are smaller than three residues, the distance is estimated by shortest path distance. According to the property of protein structure, each two adjacent C_α atoms have the average distance of 3.8 angstrom (Reese et al., 1996). In theory, any two C_α atoms can be connected by multiple adjacent C_α atoms. When we know the number of adjacent C_α atoms on the path, we can get the path with the shortest distance of all connecting adjacent C_α pairs (Jingfen Zhang et al., 2010). This can be done on a distance matrix to fill in the small gaps since the shortest path will easily overestimate the distance if the gap is too large, especially those on the terminal end.

6.7 Model Generation

MUFOLD's model generation is mainly based on the distance matrix of C_α atoms. Side chains are added back the C_α model later. There are two methods implemented in MUFOLD for model generation: the fully extended method and the distance matrix based method.

6.7.1 Method 1: Fully Extended Model Generation

The idea of fully extended method is straightforward. It is still a template-based method and the template is extended on both sides as the final model. Fully extended method is simple, but it provides a quick and fast model generation to evaluate the template quality and it performs good if we can get good templates.

The following Figure 6.4 shows an illustration of how the fully extended modeling method works. The first blue line is the query target sequence. The second light orange line is one of the templates we find. The dark orange regions in the template, i.e. *A* and *B* are

the aligned regions to the target sequence. To get the fully extended model, the aligned part in the template is extended as far as possible on the two ends but not over the length of the target sequence. The final fully extended model will be the regions A' and B' and the sequence IDs will be re-numbered to align with the IDs in the query target sequence.

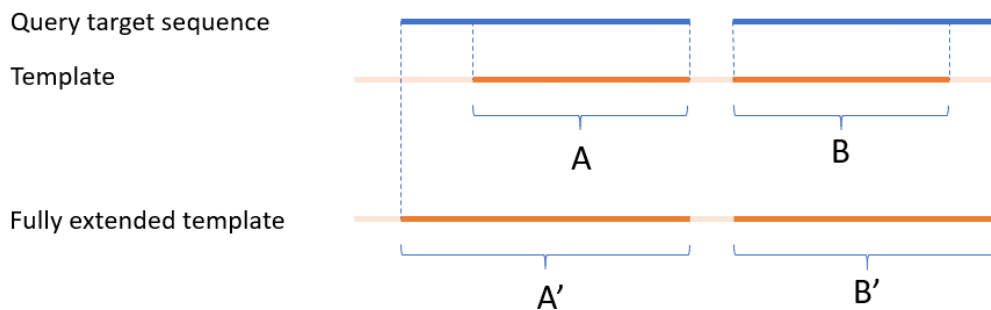


Figure 6.4 Illustration of how fully extended modeling method works.

In the fully extended modeling method, the extended area may disturb the quality of the whole structure if it is very different from the native structure of the query target. If the extended area is very long it will dominate the model as well, which makes the fully extended modeling quality poor.

6.7.2 Method 2: Distance Matrix Based Model Generation

The distance matrix is the key data structure in this method. From the previous description that all the templates are represented using distance matrix and loop modeling is done via distance matrix as well. In this section the process of generating models from distance matrix by iterative multidimensional scaling (MDS) is covered.

Iterative MDS

It is known that given a distance matrix, the 3D structure can be derived from it by MDS. However, from our experiments, sometimes MDS can distort the helix and beta sheet

units of a 3D structure. To solve this problem, an iterative process is used to keep the retain the structure of helix and beta sheets. Using iterative MDS can help to make the structure more protein-like.

The following Figure 6.5 shows the process of iterative MDS. In the figure, DM is short for distance matrix. The $DM_{alignment}$ on the upper left corner is the input distance matrix to the iterative MDS. From this $DM_{alignment}$ by applying MDS we can get a 3D model, then from this 3D model we build the distance matrix called DM_{model} . At this point if we apply MDS on DM_{model} again we may get a structure with helix and beta sheet distorted. Therefore, we add an SSE replacement step to replace the areas of the helix and beta sheet in DM_{model} with the corresponding values in $DM_{alignment}$ and the new distance matrix is called $DM_{model_sse_replaced}$, from which we can get another model by applying MDS. The helix and beta sheet areas are predicted from sequence by PSIPred. Then the RMSD of those two models are calculated. Iteration stops when the RMSD converges or up to 5 iterations totally.

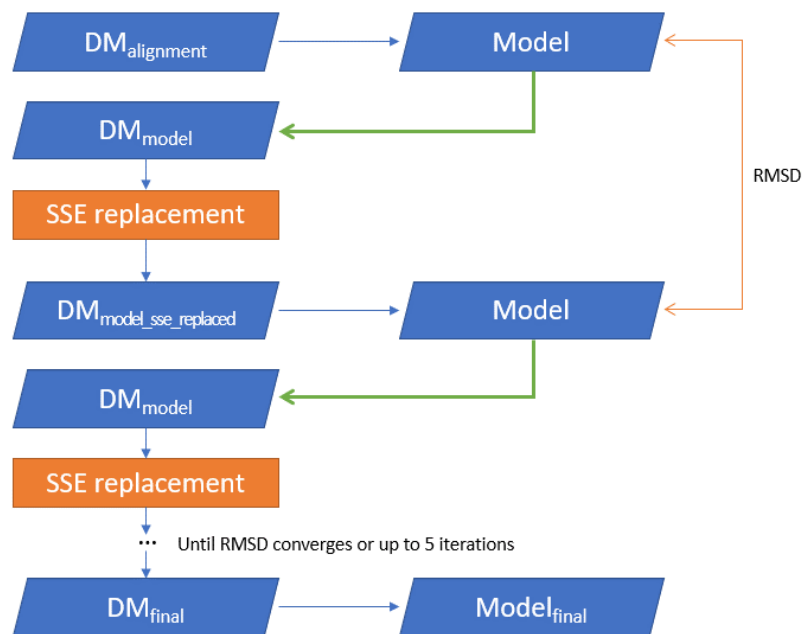


Figure 6.5 Process of iterative MDS in distance matrix based modeling.

Fixing the Mirror Case

For a distance matrix MDS can output two results in the 3D space and they are mirror to each other. In MUFOLD, it only needs the one that keeps the global topology of the template used for modeling, rather than the mirror one. To solve this problem, we introduce the determinant of orthogonal matrix of the template's coordinates and MDS output's coordinates. The following shows the process of our mirror fixing algorithm.

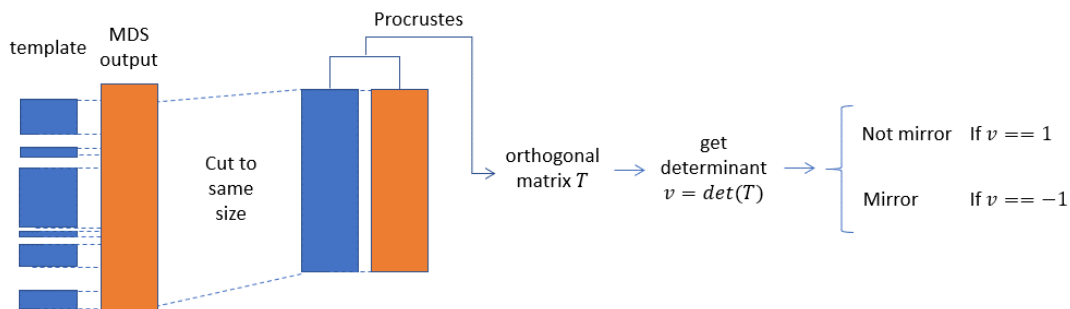


Figure 6.6 The process of fixing mirror case.

In the above figure, the coordinates from template and from MDS output will be aligned and cut to the same size. Then after doing Procrustes we can get the orthogonal matrix T . We get the determinant $v = \det(T)$. If v equals to 1 then this MSD output is not a mirror, otherwise, it is mirror and we need to switch the x and z axes.

Building Full Atom Model

Since MUFOLD does all the operations using the distance matrix of C_α atoms, the generated model is coarse grained C_α model. For better further use of the model it is needed to have a full-atom model with side chains. This process is done by a third-party tool PULCHRA (Rotkiewicz & Skolnick, 2008).

6.8 MUFOLD Modules Design

MUFOLD is designed using object oriented programming (OOP) idea. The modules are showed in the following Figure 6.7. The basic data structures are all encapsulated in the Data Objects module. All other modules are decoupled and can communicate with each other with a pre-defined communication protocol. In a word, the new design make the MUFOLD platform very easy to add new tools, develop new methods, do fast development iterations, and reuse the code bases.

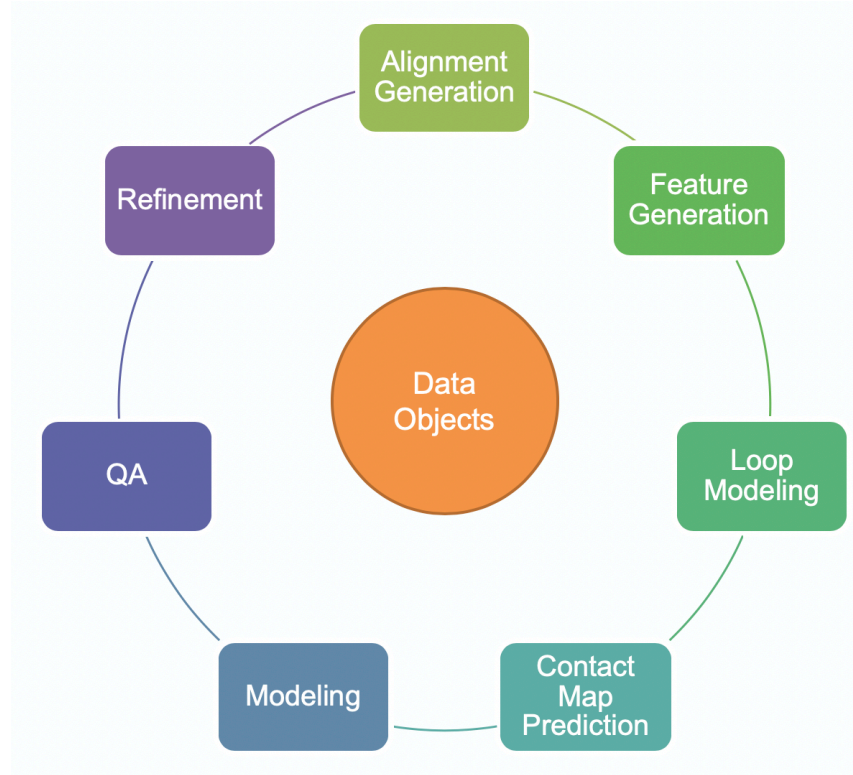


Figure 6.7 MUFOLD modules design

6.9 Model Quality Assessment

Each master alignment will generate a model candidate. In current version, we set the number of master alignment to 10 so finally 10 models will be evaluated and ranked by quality. This evaluation is done by our quality assessment module.

There are two types of model quality assessment methods in MUFOLD, the single model QA, and consensus model QA. The single model QA generates a single score for each model, and the score is used to rank. The consensus model QA generates a consensus score matrix and the average consensus score is used to rank. For CASP13, single model QA with two machine learning regressor is used: the random forest regressor and Ridge regressor. The features for each model are as follows:

1. Predicted secondary structure and solvent accessibility scores;

2. Scores generated by dfire and dDfire (Y. Yang & Zhou, 2008), DOPE (Shen & Sali, 2006), RW and RWplus (J. Zhang & Zhang, 2010), HOPP (Sims & Kim, 2006), Opus (Y. Wu, Lu, Chen, Li, & Ma, 2007), Proq2 (Uziela & Wallner, 2016), and Proq3 (Uziela, Shu, Wallner, & Elofsson, 2016);

6.10 Web Services Development

It is very important to contribute to the community by providing online services for others to use your tools. It can prove the performance of your tools and others can use your tool online directly without downloading and installing a lot of files. To use our servers, the user can provide their email address so that the user will get notification when the job is submitted and finished.

The web portal is freely available for educational purposes and can be found here: <http://dslsrv2.eecs.missouri.edu/~zlht3/>.

6.10.1 Web Server Architecture

The web services system contains three parts: the frontend webpages, the backend requests processing system, and the job management system. The overall architecture is shown in the following Figure 6.8. This architecture is universal for all our different services since we have multiple different runners to handle different jobs and it can be easily adapted to new tools. For example, the same architecture was used in CASP13 for our participation in server modeling prediction, the refinement, and the quality assessment. All of them worked very well.

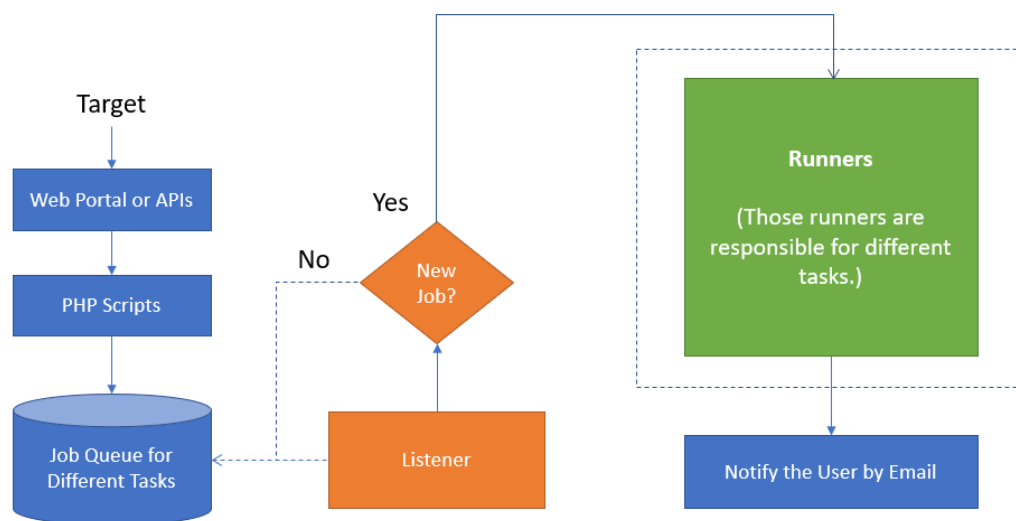


Figure 6.8 Web server architecture.

When a user submits a target query via the web page or APIs, the request will be handled by our request processing system, which is written in PHP. The request will be put into a job queue and marked the job type. If there are the same type jobs running, the new jobs will be waiting in the queue. Our job management system has a listener to the job queue to detect unfinished jobs. If there is an unfinished job to be executed, the job listener will pop it from the queue and call corresponding runner to run the job. When the job is finished by the runner, the listener will update the status of the job and send notifications to users if needed. In addition, our frontend webpages have implemented AJAX features to display the real-time status of user's job without refreshing the webpages.

6.10.2 MUFOLD SSW - Secondary Structure and Supersecondary Structure

Prediction Server

This server is to provide services for protein secondary structure, backbone torsion angle predictions, and beta-turn/gamma-turn predictions, which can provide important

information for protein 3D structure prediction and protein functions. This server has four runners integrated to provide services: MUFold-SS (Fang, Shang, & Xu, 2018a), MUFold-Angle (Fang, Shang, & Xu, 2018b), MUFold-BetaTurn (Fang, Shang, & Xu, 2018b) and MUFold-GammaTurn (Fang, Shang, & Xu, 2018a). Here, these four new software tools are made available to the community through one easy-to-use web service, called MUFOLD Secondary Structure Webserver (MUFOLD-SSW).

The first version of MUFold-SSW was released in June 2017. Since then, there have been about 100-150 submissions per month from dozens of sites around the world.

A user can submit a sequence at our web interface, and the submit-ted job will be queued on the server to run sequentially.

Input

The input to MUFold-SSW includes up to 10 sequences in the FASTA format, type of predictions (SS, TA, BT, and/or GT), and optionally user email address and job name. The following Figure 6.8 shows the job submission page. The web interface has 1) user email address, 2) the job name, 3) the job type, 4) the sequence to be submitted, and at the bottom there is a button to submit. A user can also click on an example sequence link to submit an example job or click to see an example result.

MUFOLD Servers MUFOLD_SS

MUFOLD SS and Supersecondary Structure Prediction

Protein structure prediction is very heated research area in computational bioinformatics field. A Faster and more accurate protein secondary structure prediction tool can provide important support for ab initio and homology modelling protein 3D structure prediction. The protein secondary structure is acting as a bridge that link the sequence and 3D structure.

Submit New Job
My Job Status
Software Download
Documentation
Contact

Email address (Optional, later you can use either your email or job ID to track your jobs) 1
Enter email

Job Name (Optional) 2
Enter target name

Job Type 3
 Secondary Structure (3-states and 8-states)
 Psi-phi Angles
 Beta Turn
 Gamma Turn

Input Sequence (up to 10 sequences, must be in FASTA format) [paste example] [See example results] 4
Enter sequence

Submit

Figure 6.9 MUFOLD_SSW server job submission page.

Output

After a job is submitted, it will be assigned a unique job ID and a web URL to track the job status. If the user submits a valid email address, they will receive the job running status link in the email and use their email to track the job. The following Figure 6.10 shows an example email notification when the job is submitted and finished.

Your MUFOLD_SSW job ss_5c88342b9b47c is submitted.



MUFOLD@dsisrv2.eecs.missouri.edu <MUFOLD@dsisrv2.eecs.missouri.edu>
Li, Zhaoyu (MU-Student)
Tuesday, March 12, 2019 at 5:35 PM
[Show Details](#)

Hi,

We have received your submission.

You can check the real time job status in the following link:

http://dsisrv2.eecs.missouri.edu/~zih3/ss/results/ss_5c88342b9b47c

Thank you!

Your MUFOLD_SSW job ss_5c88342b9b47c is finished.



MUFOLD@dsisrv2.eecs.missouri.edu <MUFOLD@dsisrv2.eecs.missouri.edu>
Li, Zhaoyu (MU-Student)
Tuesday, March 12, 2019 at 5:39 PM
[Show Details](#)

Hi,

Your job is finished.

Check out the job results in the following link:

http://dsisrv2.eecs.missouri.edu/~zih3/ss/results/ss_5c88342b9b47c

Thank you!

Figure 6.10 The email notifications when a job is submitted and finished.

On the job running status webpage, the real-time job status and job running logs are displayed. The following Figure 6.11 shows the job status page. The web interface shows 1) the job summary, 2) real-time job status, and 3) real-time job running logs.

MUFOLD Servers MUFOLD_S5

MUFOLD SS and Supersecondary Structure Prediction

- Submit New Job
- My Job Status**
- Software Download
- Documentation
- Contact

Job ID:	ss_5c04e6395feae
Link to This Job Results:	http://dsisrv2.eecs.missouri.edu/~zih3/ss/results/ss_5c04e6395feae
Target Name:	test_1
Email:	Not_provided_5c04e6395fe37 Click here to track all your jobs using this email
Submitted Time:	2018-12-03 03:15:53
Finished Time:	Waiting...
Your job types:	SS

Current status: **Running...**

Running Logs Results Downloads Results File

The following logs are refreshed every 5 seconds automatically.

[2018-12-03 03:15:53] Your job is pending now...
[2018-12-03 02:15:55] Your job is running...
[2018-12-03 02:15:55] Predicting Secondary Structure...

Figure 6.11 MUFOLD_SSW server job running status page.

When the job is finished, the user will receive another email notification with the link to the job result page. The result webpage includes a job summary, detailed prediction results, distributions and histograms of the results, and links to download all the result files. The following Figure 6.12 shows the job result page. The web interface shows 1) the navigation menu, 2) job summary, 3) job status, 4) the results navigation tabs to show running logs, results, and results download links, 5) the prediction results, 6) the distributions and histograms of all predictions, and 7) the raw text output.

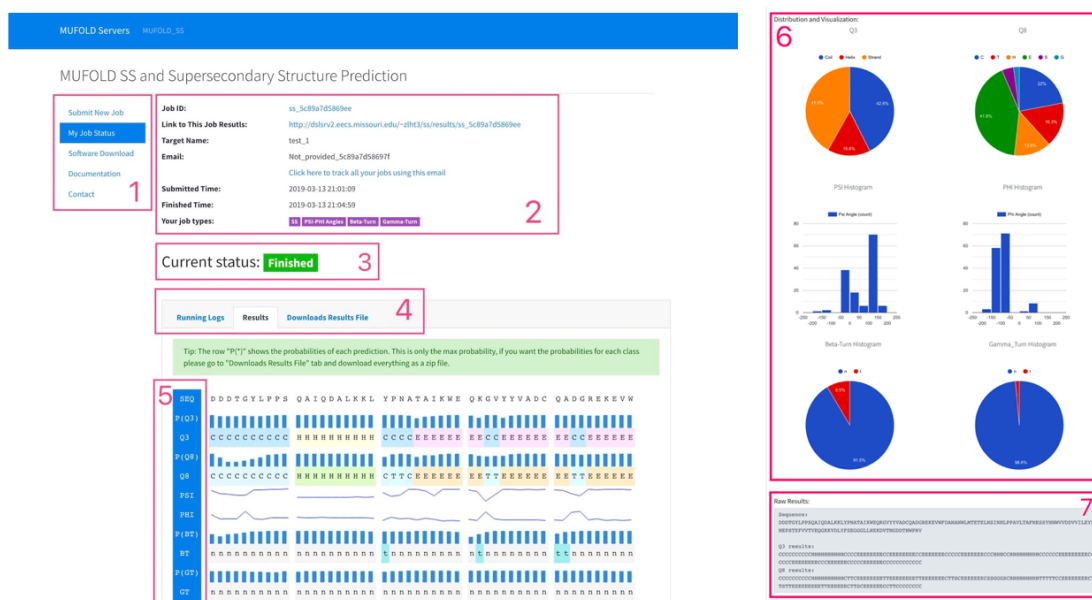


Figure 6.12 MUFOLD_SSW server job result page.

6.10.3 MUFOLD 3D Structure Prediction Server

This server is the one used to participate in CASP13 competition for 3D structure prediction category. It provides several APIs for CASP official use and submits modeling results to CASP server automatically without any human interference. This server will receive the target name, target sequence, and submission email via a POST request. Our backend will process the request and save the job information to the queue and send

confirmation back to CASP. Our job management system is monitoring the job queue and once there is a job in pending, it will call our prediction toolkit MUFOLD to generate the top 5 models. After MUFOLD is finished, the job management system will reformat those models to the standard CASP format and send them to the submission email address. This server worked smoothly during the CASP13 competition and never missed any single target query.

Currently this server can only accept requests by POST API call. We will open the access to the public with a frontend user interface once in the future.

6.10.4 MUFOLD Contact Prediction Server

This is the server used to participate in the CASP13 competition for the residue-residue contact prediction category. It provides several APIs for CASP official to use and submits the predicted results to the CASP server automatically. It works in the same way as our previous two servers. When a new job is submitted by API, our backend will process it and put it into the job queue. The job management system will read the job queue and run any pending jobs. When the job is finished, the job management system will reformat the results and submit it to the specified CASP submission email address. The same as our modeling server, this server worked very well and smoothly during the CASP13 competition without human interference.

The same as 3D structure prediction server, currently this server can only accept requests by POST API call. We will open the access to the public with a frontend user interface once in the future.

6.10.5 Server Usefulness

Since the release of our server, there are 8,351 different submissions from 1,521 users from all over the world until the end of April 2020. See the following Figure 6.13 for a world map of visitor from Google Analytics.

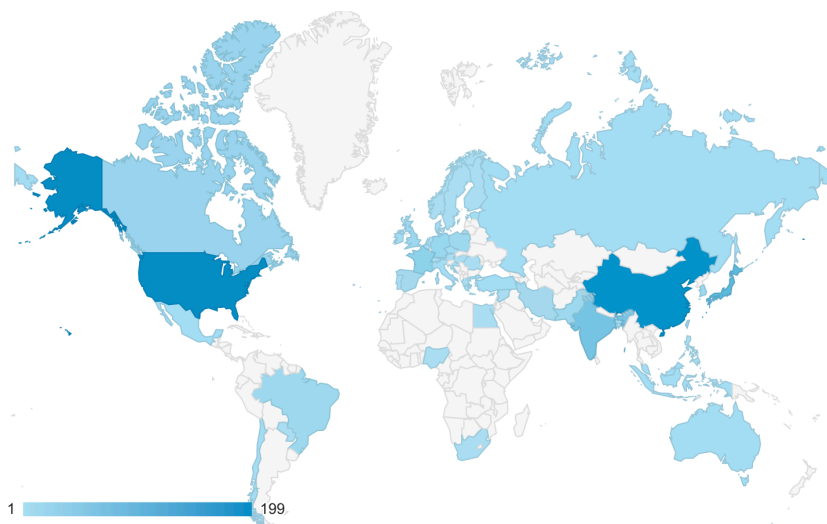


Figure 6.13 World map of MUFOLD server visitors.

We also keep receiving feedbacks from the users and use their feedbacks to make the server more stable. For example, the following Figure 6.14 shows the most recent email we received from one of our users. We are very proud that our server can make contributions to the community and their feedbacks are also very helpful to use to improve the server.

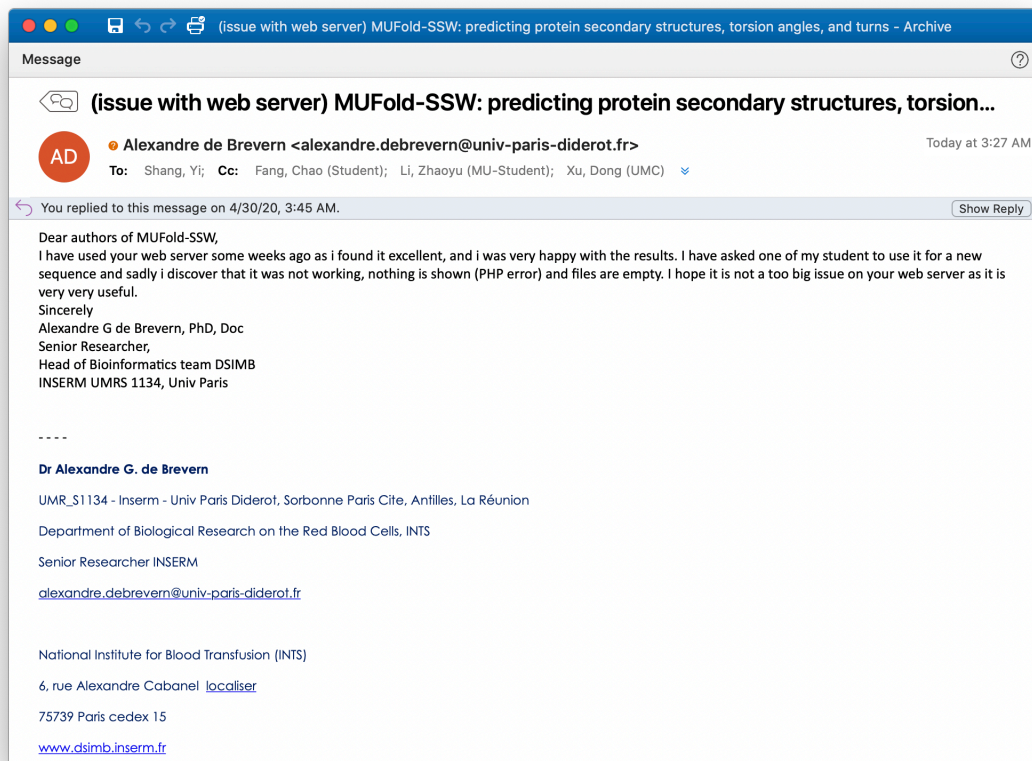


Figure 6.14 Recent feedback received from our server user.

6.11 Summary

In this chapter, the design and the development of our comprehensive protein prediction platform, the MUFOLD, is introduced. It contains several tools for different type of protein prediction tasks, such as secondary structure and supersecondary structure prediction, loop modeling toolkit, template-based 3D structure prediction, contact map prediction, and quality assessment toolkit. The system pipeline is written in object-oriented programming language C++ and Python to decouple each module and to make it easier for new tools and algorithms to be tested in the whole pipeline in a fast iteration.

The web portal is also introduced to provide several online services to the community. The web service system consists of three major parts: the frontend, the backend, and the job management system. The frontend provides a user-friendly and easy to use web interface for users to submit their jobs. The backend handles all web and API requests and manage the job queue on the server side. The job management system is responsible to arrange the executions for each job, provide real-time status, and generate results report when the job is finished. This architecture can be easily adapted to other web services in the future. More specifically, the web services system provides three services: the secondary and supersecondary structure prediction, i.e. angles, beta turn, and gamma turn; the 3D structure modeling service; and the contact map prediction. The first one is publicly available now and the last two are specifically for CASP competition purpose by API call. In the near future we will open the access to the last two servers to the public. More services are on the way and the goal is to provide a unified web portal of all our tools for the community.

CHAPTER 7. SUMMARY AND FUTURE WORK

7.1 Summary

In this thesis, two deep learning based methods for protein loop modeling and contact map prediction, and the development of two projects, the MUFOLD platform and the web portal for our tools, have been proposed.

For the loop modeling problem, MUFOLD_LM method is the first successful GAN application in bioinformatics. The network consists of a generator and an adversarial discriminator. The generator generates the predictions of the missing region based on the existing region's context, and the discriminator try to sharp the generated distance map of the missing region to make it more protein like. The predicted distance map is converted back to 3D structure using MDS algorithm. It is proved that the GAN architecture improves the quality of the predictions and significantly reduce the standard deviation for all predictions. The feasibility has been verified so that the similar GAN architecture could be applied to other bioinformatics problems as well.

For the contact map prediction, progress of improving the performance of our baseline model in CASP13 gradually has been shown. Different ways of getting contact map predictions are explored. The current best proposed model is a two-stages multi-branch network to combine the distance map prediction and the binary contact map prediction to fully utilize the information from the features set. We have been in the top range of all groups that are using the almost same features set and we believe by integrating more advanced features, such as the raw score, and by tailoring the network structure we can keep improving the performance of our model.

For the predicted contact map refinement, TPCref employed new methods to use protein templates information and contact predictions on these templates to improve prediction accuracy on the original target protein. It is general and can be applied over any existing method. Significant improvement has been shown in our experimental results.

The development and functions of our comprehensive protein structure prediction platform, MUFOLD has also been shown. It has been refactored using object-oriented programming language to decouple multiple modules. It provides us the opportunities to try new ideas and tools and get feedback in fast pace. The basic template-based pipeline has been finished and multiple tools have been integrated including the secondary structure and supersecondary structure predictor, the 3D structure modeling using different modeling methods, the contact map predictor, and the model quality assessment tools. In addition, it is very important to have our tools being tested by others and make contributions to the community as well. A web services system including the backend, frontend, and the job management system, has been designed and implemented. Users can submit their jobs online or via APIs request and get notified when the job is finished, without downloading and installing a ton of tools, databases, and third-party dependencies on their own computer. It can be easily adapted to our new tools in the future since the well-designed system architecture and the easy-to-use user interfaces.

Finally, here is the list of publications related to the work in this research (in chronological order):

1. Chao Fang*, Zhaoyu Li* (co-first author), Dong Xu, Yi Shang. "MUFold-SSW: A New Webserver for Predicting Protein Secondary Structures, Torsion Angles, and Turns." *Bioinformatics* 36, no. 4 (2020): 1293-1295.

2. Ruffolo, Jeffrey, Zhaoyu Li, and Yi Shang. "MUFold-Contact and TPCref: New Methods for Protein Structure Contact Prediction and Refinement." In 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), pp. 90-93. IEEE, 2019.
3. Wang, Wenbo, Zhaoyu Li, Junlin Wang, Dong Xu, and Yi Shang. "PSICA: a fast and accurate web service for protein model quality analysis." *Nucleic acids research* 47, no. W1 (2019): W443-W450.
4. Zhaoyu Li, Son Nguyen, Dong Xu, Yi Shang. "Protein Loop Modeling Using Deep Generative Adversarial Network." In Tools with Artificial Intelligence (ICTAI), 2017 IEEE 29th International Conference IEEE, 2017
5. Wang, Junlin, Zhaoyu Li, and Yi Shang. "New Deep Neural Networks for Protein Model Evaluation." In 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI), pp. 309-313. IEEE, 2017.
6. Son Nguyen, Zhaoyu Li, Yi Shang. "Deep Networks and Continuous Distributed Representation of Protein Sequences for Protein Quality Assessment." In Tools with Artificial Intelligence (ICTAI), 2017 IEEE 29th International Conference IEEE, 2017.
7. Son Nguyen, Zhaoyu Li, Dong Xu, and Yi Shang. "New Deep Learning Methods for Protein Loop Modeling." *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (2017).

7.2 Future Work

There are some further experiments ongoing as the future work in terms of loop modeling, contact map prediction, etc.

7.2.1 Loop Modeling

As the first successful application of GAN in bioinformatics, our loop modeling work opens a door for many other GAN applications in analysis and prediction problems of biological sequences structures, such as protein contact prediction and 3D genome structure prediction (Z. Li et al., 2017). Future work will be done to make our method more general to all problems. Our GAN study also has room for improvement. In our method, the loop region must be in the middle of the input subsequence. In future work, an input parameter indicating the loop region can be applied to solve this problem. In addition, the training for GAN sometimes is not stable. More advanced GAN training methods can improve the training stability.

7.2.2 Contact Map Prediction

Using Generative Adversarial Network

Since GAN has been proved to be helpful in the loop modeling problem to have a better distance map prediction, it is worth to try GAN for contact map prediction. Our two-stage multi-branch network uses predicted distance map from stage 1 as intermediate features and GAN can be used in stage 1, it could improve the final contact map prediction accuracy.

The intuitions behind the idea that GAN can improve the protein's complete distance map are as follows. First, distance map is not random value matrix. Each distance has relationship with other distances and all the distances must be comply with some

distributions or patterns to be a protein-like distance map. The GAN can be used to learn this distribution inside the proteins. Second, the values in the distance map can range from 0 to a very large value, and the maximum value for each protein is different because it depends on the size of protein in the 3D space. The traditional deep neural network can predict distance, but it may be scaled down or up. The GAN can be used to learn the right scaling factors in a protein's distance map to make sure the predicted distance map is in the right scale. Third, we trained the distance map prediction network using mean square error (MSE) loss, which has been proved to get blurry results in the prediction for images. It is intended to get a minimum loss for all pixels in the image so the sharpness of pixels could be sacrificed. It is the same situation for distance map prediction. The MSE loss will make the network try to predict values that have a minimum difference on average, which makes the predicted distance map blurry. The GAN has been proved to be able to solve this problem in image inpainting problem. The filled in image patch looks sharper than without using GAN.

One problem to be resolved is our network take various size of proteins as input and predict the same size output. However, in a traditional GAN the discriminator network will have fully connected layers at the end and generate a true or false value, which means the input size to the discriminator network should be fixed. This is also a problem for some image inpainting problem if the input images have different sizes. PatchGAN has been proposed to solve this problem in pix2pix (Isola, Zhu, Zhou, & Efros, 2017), and this idea was first introduced in paper (C. Li & Wand, 2016). The basic idea of PatchGAN is to use a sliding window to get a fixed size subset of the whole output of generator so that the discriminator can have a fixed size input no matter what the size of generator's output. The

following Figure 7.1 shows the basic idea of using PatchGAN in our distance map prediction network.

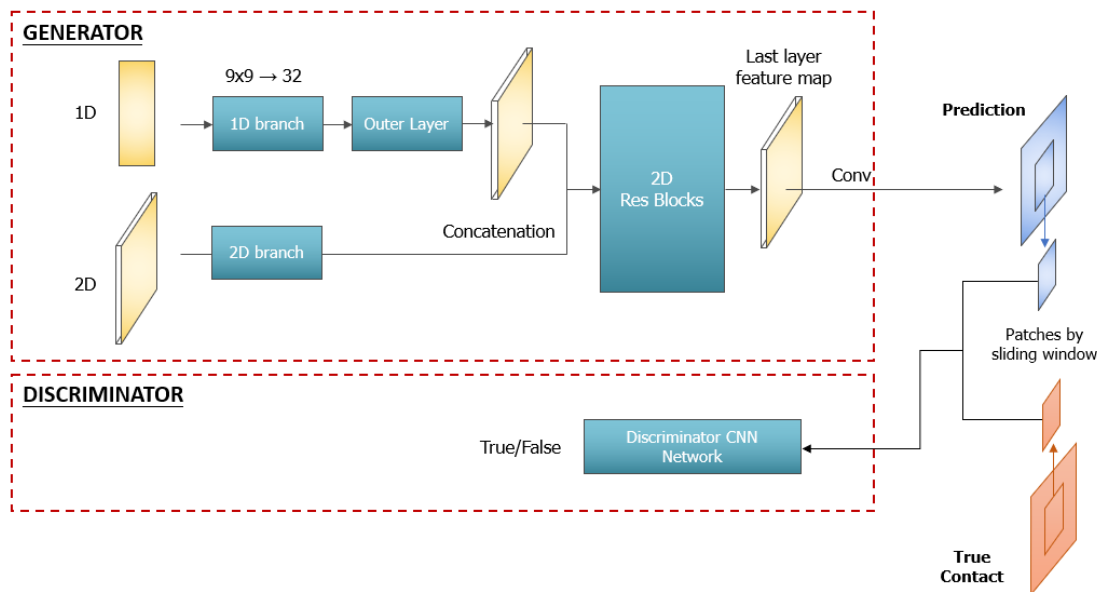


Figure 7.1 The idea of using PatchGAN in distance map prediction network.

Template Assisted Contact Map Prediction

Current contact map prediction methods are all using sequence database to get the multiple sequence alignments. The sequence database can give a large number of homologous sequences but most of them don't have known structures. The traditional template-based 3D structure prediction methods rely on the known structures, which inspires us that if we import known structures in the contact map prediction, we would get better results.

To prove our idea is feasible, some experiments have been done. For the CASP13 testing set, there are structure predictions from other servers available, and the good models are almost all based on templates. If we derive the contact prediction from those model

predictions, it is like we use the information from the templates. If the results are not too bad, it means the template information can help.

The following table shows the testing results using CASP13 targets in terms of Long-range top L/5. The top L/5 predictions are selected by distance. Best-In-Pool means we use the best predicted model among all the server models to calculate contact map. Best-In-150 means we use the best predicted model among the set-150 provided by CASP for quality assessment purpose. Accordingly, the Best-In-20 means among the set-20 provided by CASP.

Table 7.1 The average accuracy for contact derived from best predicted model in the pool in CASP13.

	Best-In-Pool	Best-In-150	Best-In-20
Average accuracy	70.61%	70.61%	66.14%

The accuracy for Best-In-Pool is as high as 70.61%, which shows the advantage of using templates for contact prediction. If the same templates that are used for modeling can be used in the contact map prediction, the final prediction accuracy could be significantly improved. The future work is to figure out a way to integrate the template information in the process of contact map prediction.

DeepCov Raw Score for Contact Map Prediction

DeepCov is a new published contact map predictor using the raw covariance scores as input to a deep neural network to do contact map prediction (D. T. Jones & Kandathil, 2018). In this paper they proposed the raw score that is derived from the multiple sequence alignment directly and this is the only features used. The raw score has been proved to carry more information and can improve the performance. After CASP13, it is also noticed

that many top groups were using the raw score as their inputs as well and they all performed very well.

For the raw score, each residue pair has a covariance matrix with dimension of 21×21 . For a protein of length L , the raw score feature has the size of $L \times L \times 21 \times 21$, which is very large. In the future work, we will add this feature to our current pipeline and figure out a way to reduce the dimension.

7.2.3 Predicted Contact Map Refinement

TPCref offers a highly customizable framework for contact refinement, which can be tuned according to the purposes of its user. In the experiments described in Chapter 5, several parameters were fixed in the interest of evaluating a variety of methods in a reasonable amount of time. However, there are several parameters (e.g. template clustering method, number of templates, template filter weighting, etc.) which might be varied in order to achieve improved performance. For example, utilizing a greater number of templates should provide more information and result in further improvements through refinement. Similarly, a more sophisticated template weighting scheme might better incorporate the information contained in the template filters.

A notable drawback to TPCref is the substantial increase in computation time required for its application. To use the method as described above involves prediction of contacts for an additional 20 sequences. For methods with long runtimes, this may be prohibitive. However, in such cases some refinement could still be achieved with a reduced number of templates, in order to reduce computational overhead.

7.2.4 A Unified Web Portal for All Our Tools

Since a web services system including the frontend, backend, and job management system has been implemented, it is essential to make it a unified web portal for all our current and future tools. Currently, the MUFOLD_SSW server is publicly available for the community, the 3D structure modeling server and the contact map prediction server are available for CASP competition purpose by providing APIs. In order to provide a unified web portal, in the near future we will open access to all our tools via the user-friendly and easy to use web interfaces. At the same time, APIs call will be kept in case others want to use it in their own scripts and for the future CASP competitions. By the unified web portal, we believe we can make our tools more accessible and make the community much stronger.

VITA

Zhaoyu Li is currently a Ph.D. candidate in Distributed and Intelligent Computing Lab at the Electrical Engineering and Computer Science (EECS) Department, University of Missouri, under the supervision of Professor Yi Shang.

He obtained his M.S. degree in Computer Science at University of Missouri and obtained his B.S. degree in Software Engineering at Beijing Jiaotong University.

His research focuses on developing novel deep learning architectures for computational biology images and models. The techniques that he actively uses in his research involve deep learning, machine learning, artificial intelligence, optimization methods, data mining, data structure and algorithms, and statistical analysis.

REFERENCE

- Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., . . . Devin, M. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.
- Adhikari, B., & Cheng, J. (2018). CONFOLD2: improved contact-driven ab initio protein structure modeling. *BMC bioinformatics*, *19*(1), 22. doi:10.1186/s12859-018-2032-6
- Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. (1997). Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*, *25*(17), 3389-3402. doi:10.1093/nar/25.17.3389
- Anfinsen, C. B. (1973). Principles that govern the folding of protein chains. *Science*, *181*(4096), 223-230.
- Baldassi, C., Zamparo, M., Feinauer, C., Procaccini, A., Zecchina, R., Weigt, M., & Pagnani, A. (2014). Fast and Accurate Multivariate Gaussian Modeling of Protein Families: Predicting Residue Contacts and Protein-Interaction Partners. *PLoS One*, *9*(3), e92721. doi:10.1371/journal.pone.0092721
- Bhattacharya, D., Nowotny, J., Cao, R., & Cheng, J. (2016). 3Drefine: an interactive web server for efficient protein structure refinement. *Nucleic Acids Res*, *44*(W1), W406-409. doi:10.1093/nar/gkw336
- Brunger, A. T. (2007). Version 1.2 of the Crystallography and NMR system. *Nat Protoc*, *2*(11), 2728-2733. doi:10.1038/nprot.2007.406
- Brunger, A. T., Adams, P. D., Clore, G. M., DeLano, W. L., Gros, P., Grosse-Kunstleve, R. W., . . . Warren, G. L. (1998). Crystallography & NMR system: A new software suite for macromolecular structure determination. *Acta Crystallogr D Biol Crystallogr*, *54*(Pt 5), 905-921. doi:10.1107/s0907444998003254
- Cheng, J., & Baldi, P. (2007). Improved residue contact prediction using support vector machines and a large feature set. *BMC bioinformatics*, *8*(1), 113.
- Choi, Y., & Deane, C. M. (2010). FREAD revisited: Accurate loop structure prediction using a database search algorithm. *Proteins-Structure Function and Bioinformatics*, *78*(6), 1431-1440. Retrieved from <Go to ISI>://WOS:000276369700007
http://onlinelibrary.wiley.com/store/10.1002/prot.22658/asset/22658_ftp.pdf?v=1&t=itvyf8k4&s=d743eeead7965c32df4b46f547ab2bfae1452de
- Chys, P., & Chacon, P. (2013). Random Coordinate Descent with Spinor-matrices and Geometric Filters for Efficient Loop Closure. *J Chem Theory Comput*, *9*(3), 1821-1829. doi:10.1021/ct300977f
- Collobert, R., & Weston, J. (2008). *A unified architecture for natural language processing: Deep neural networks with multitask learning*. Paper presented at the Proceedings of the 25th international conference on Machine learning.
- de Bakker, P. I., DePristo, M. A., Burke, D. F., & Blundell, T. L. (2003). Ab initio construction of polypeptide fragments: Accuracy of loop decoy discrimination by

- an all-atom statistical potential and the AMBER force field with the Generalized Born solvation model. *Proteins*, 51(1), 21-40. doi:10.1002/prot.10235
- Eickholt, J., & Cheng, J. (2012). Predicting protein residue–residue contacts using deep networks and boosting. *Bioinformatics*, 28(23), 3066-3072.
- Ekeberg, M., Lökvist, C., Lan, Y., Weigt, M., & Aurell, E. (2013). Improved contact prediction in proteins: Using pseudolikelihoods to infer Potts models. *Physical Review E*, 87(1), 012707. doi:10.1103/PhysRevE.87.012707
- Fang, C., Shang, Y., & Xu, D. (2018a). Improving Protein Gamma-Turn Prediction Using Inception Capsule Networks. *Scientific Reports*, 8(1), 15741. doi:10.1038/s41598-018-34114-2
- Fang, C., Shang, Y., & Xu, D. (2018b). *MUFold-BetaTurn: A Deep Dense Inception Network for Protein Beta-Turn Prediction*.
- Fang, C., Shang, Y., & Xu, D. (2018a). MUFOLD-SS: New deep inception-inside-inception networks for protein secondary structure prediction. *Proteins*, 86(5), 592-598. doi:10.1002/prot.25487
- Fang, C., Shang, Y., & Xu, D. (2018b). Prediction of Protein Backbone Torsion Angles Using Deep Residual Inception Neural Networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1-1. doi:10.1109/TCBB.2018.2814586
- Fink, A., Kosecoff, J., Chassin, M., & Brook, R. H. (1984). Consensus methods: characteristics and guidelines for use. *Am J Public Health*, 74(9), 979-983.
- Fiser, A. (2010). Template-based protein structure modeling. *Methods Mol Biol*, 673, 73-94. doi:10.1007/978-1-60761-842-3_6
- Fiser, A., Do, R. K., & Sali, A. (2000). Modeling of loops in protein structures. *Protein Sci*, 9(9), 1753-1773. doi:10.1110/ps.9.9.1753
- Fowler, J. E. (2005). The redundant discrete wavelet transform and additive noise. *IEEE Signal Processing Letters*, 12(9), 629-632. doi:10.1109/LSP.2005.853048
- Ginalski, K. (2006). Comparative modeling for protein structure prediction. *Curr Opin Struct Biol*, 16(2), 172-177. doi:10.1016/j.sbi.2006.02.003
- Glorot, X., Bordes, A., & Bengio, Y. (2011). *Deep Sparse Rectifier Neural Networks*. Paper presented at the Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research. <http://proceedings.mlr.press>
- Godzik, A., & Sander, C. (1989). Conservation of residue interactions in a family of Ca-binding proteins. *Protein Eng*, 2(8), 589-596. Retrieved from <https://www.ncbi.nlm.nih.gov/pubmed/2813336>
- Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). *Generative Adversarial Nets*. Paper presented at the NIPS.
- He, B., Mortuza, S., Wang, Y., Shen, H.-B., & Zhang, Y. (2017). NeBcon: protein contact map prediction using neural network training coupled with naïve Bayes classifiers. *Bioinformatics*, 33(15), 2296-2306.
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*.

- He, Z., Zhang, C., Xu, Y., Zeng, S., Zhang, J., & Xu, D. (2014). MUFOLD-DB: a processed protein structure database for protein structure prediction and analysis. *BMC Genomics*, *15 Suppl 11*, S2. doi:10.1186/1471-2164-15-s11-s2
- Heo, L., Park, H., & Seok, C. (2013). GalaxyRefine: Protein structure refinement driven by side-chain repacking. *Nucleic Acids Res*, *41*(Web Server issue), W384-388. doi:10.1093/nar/gkt458
- Hildebrand, P. W., Goede, A., Bauer, R. A., Gruening, B., Ismer, J., Michalsky, E., & Preissner, R. (2009). SuperLooper--a prediction server for the modeling of loops in globular and membrane proteins. *Nucleic Acids Res*, *37*(Web Server issue), W571-574. doi:10.1093/nar/gkp338
- Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A., Jaitly, N., . . . Kingsbury, B. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, *29*(6), 82-97. doi:10.1109/MSP.2012.2205597
- Ioffe, S., & Szegedy, C. (2015). *Batch normalization: accelerating deep network training by reducing internal covariate shift*. Paper presented at the Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37, Lille, France.
- Isola, P., Zhu, J., Zhou, T., & Efros, A. A. (2017, 21-26 July 2017). *Image-to-Image Translation with Conditional Adversarial Networks*. Paper presented at the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
- Johnson, M. S., Srinivasan, N., Sowdhamini, R., & Blundell, T. L. (1994). Knowledge-based protein modeling. *Crit Rev Biochem Mol Biol*, *29*(1), 1-68. doi:10.3109/10409239409086797
- Jones, D. T. (1999). Protein secondary structure prediction based on position-specific scoring matrices. *J Mol Biol*, *292*(2), 195-202. doi:10.1006/jmbi.1999.3091
- Jones, D. T., Buchan, D. W., Cozzetto, D., & Pontil, M. (2012). PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. *Bioinformatics*, *28*(2), 184-190. doi:10.1093/bioinformatics/btr638
- Jones, D. T., & Kandathil, S. M. (2018). High precision in protein contact prediction using fully convolutional neural networks and minimal sequence features. *Bioinformatics*, *34*(19), 3308-3315. doi:10.1093/bioinformatics/bty341
- Jones, D. T., Singh, T., Kosciolk, T., & Tetchner, S. (2014). MetaPSICOV: combining coevolution methods for accurate prediction of contacts and long range hydrogen bonding in proteins. *Bioinformatics*, *31*(7), 999-1006.
- Jones, D. T., Singh, T., Kosciolk, T., & Tetchner, S. (2015). MetaPSICOV: combining coevolution methods for accurate prediction of contacts and long range hydrogen bonding in proteins. *Bioinformatics*, *31*(7), 999-1006. doi:10.1093/bioinformatics/btu791
- Kaján, L., Hopf, T. A., Kalaš, M., Marks, D. S., & Rost, B. (2014). FreeContact: fast and free software for protein contact prediction from residue co-evolution. *BMC bioinformatics*, *15*(1), 85. doi:10.1186/1471-2105-15-85
- Kamisetty, H., Ovchinnikov, S., & Baker, D. (2013). Assessing the utility of coevolution-based residue-residue contact predictions in a sequence- and structure-rich era.

- Proceedings of the National Academy of Sciences*, 110(39), 15674-15679. doi:10.1073/pnas.1314045110
- Kinch, L. N., Li, W., Monastyrskyy, B., Kryshtafovych, A., & Grishin, N. V. (2016). Evaluation of free modeling targets in CASP11 and ROLL. *Proteins*, 84 Suppl 1, 51-66. doi:10.1002/prot.24973
- Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.
- Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012). *ImageNet Classification with Deep Convolutional Neural Networks*. Paper presented at the Neural Information Processing Systems, Lake Tahoe, Nevada.
- Levefelt, C., & Lundh, D. (2006). A fold-recognition approach to loop modeling. *J Mol Model*, 12(2), 125-139. doi:10.1007/s00894-005-0003-0
- Li, C., & Wand, M. (2016). *Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks*, Cham.
- Li, J., Bhattacharya, D., Cao, R., Adhikari, B., Deng, X., Eickholt, J., & Cheng, J. (2014). The MULTICOM protein tertiary structure prediction system. *Methods Mol Biol*, 1137, 29-41. doi:10.1007/978-1-4939-0366-5_3
- Li, Y., Hu, J., Zhang, C., Yu, D. J., & Zhang, Y. (2019). ResPRE: high-accuracy protein contact prediction by coupling precision matrix with deep residual neural networks. *Bioinformatics*, 35(22), 4647-4655. doi:10.1093/bioinformatics/btz291
- Li, Z., Nguyen, S. P., Xu, D., & Shang, Y. (2017, 6-8 Nov. 2017). *Protein Loop Modeling Using Deep Generative Adversarial Network*. Paper presented at the 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI).
- Liu, T., Wang, Y., Eickholt, J., & Wang, Z. (2016). Benchmarking Deep Networks for Predicting Residue-Specific Quality of Individual Protein Models in CASP11. *Sci Rep*, 6, 19301. doi:10.1038/srep19301
- Lopez-Blanco, J. R., Canosa-Valls, A. J., Li, Y. H., & Chacon, P. (2016). RCD plus : Fast loop modeling server. *Nucleic Acids Research*, 44(W1), W395-W400. doi:10.1093/nar/gkw395
- Ma, J., Peng, J., Wang, S., & Xu, J. (2012). A conditional neural fields model for protein threading. *Bioinformatics (Oxford, England)*, 28(12), i59-i66. doi:10.1093/bioinformatics/bts213
- Masci, J., Meier, U., Cireşan, D., & Schmidhuber, J. (2011, 2011//). *Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction*. Paper presented at the Artificial Neural Networks and Machine Learning – ICANN 2011, Berlin, Heidelberg.
- Michalsky, E., Goede, A., & Preissner, R. (2003). Loops In Proteins (LIP)--a comprehensive loop database for homology modelling. *Protein Eng*, 16(12), 979-985. doi:10.1093/protein/gzg119
- Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., & Khudanpur, S. (2010). *Recurrent neural network based language model*. Paper presented at the INTERSPEECH.
- Monastyrskyy, B., D'Andrea, D., Fidelis, K., Tramontano, A., & Kryshtafovych, A. (2014). Evaluation of residue-residue contact prediction in CASP10. *Proteins*, 82 Suppl 2, 138-153. doi:10.1002/prot.24340

- Monastyrskyy, B., D'Andrea, D., Fidelis, K., Tramontano, A., & Kryshchak, A. (2016). New encouraging developments in contact prediction: Assessment of the CASP11 results. *Proteins*, *84 Suppl 1*, 131-144. doi:10.1002/prot.24943
- Monastyrskyy, B., Fidelis, K., Tramontano, A., & Kryshchak, A. (2011). Evaluation of residue-residue contact predictions in CASP9. *Proteins*, *79 Suppl 10*, 119-125. doi:10.1002/prot.23160
- Morcos, F., Pagnani, A., Lunt, B., Bertolino, A., Marks, D. S., Sander, C., . . . Weigt, M. (2011). Direct-coupling analysis of residue coevolution captures native contacts across many protein families. *Proceedings of the National Academy of Sciences*, *108*(49), E1293-E1301. doi:10.1073/pnas.1111471108
- Olson, M. A., Feig, M., & Brooks, C. L., 3rd. (2008). Prediction of protein loop conformations using multiscale modeling methods with physical energy scoring functions. *J Comput Chem*, *29*(5), 820-831. doi:10.1002/jcc.20827
- Park, H., Lee, G. R., Heo, L., & Seok, C. (2014). Protein loop modeling using a new hybrid energy function and its application to modeling in inaccurate structural environments. *PLoS One*, *9*(11), e113811. doi:10.1371/journal.pone.0113811
- Park, H., & Seok, C. (2012). Refinement of unreliable local regions in template-based protein models. *Proteins*, *80*(8), 1974-1986. doi:10.1002/prot.24086
- Reese, M. G., Lund, O., Bohr, J., Bohr, H., Hansen, J. E., & Brunak, S. (1996). Distance distributions in proteins: a six-parameter representation. *Protein Engineering, Design and Selection*, *9*(9), 733-740. doi:10.1093/protein/9.9.733
- Remmert, M., Biegert, A., Hauser, A., & Söding, J. (2011). HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nature Methods*, *9*, 173. doi:10.1038/nmeth.1818
- <https://www.nature.com/articles/nmeth.1818#supplementary-information>
- Rohl, C. A., Strauss, C. E. M., Misura, K. M. S., & Baker, D. (2004). Protein Structure Prediction Using Rosetta. In *Methods in Enzymology* (Vol. 383, pp. 66-93): Academic Press.
- Rotkiewicz, P., & Skolnick, J. (2008). Fast procedure for reconstruction of full-atom protein models from reduced representations. *J Comput Chem*, *29*(9), 1460-1465. doi:10.1002/jcc.20906
- Samish, I., Bourne, P. E., & Najmanovich, R. J. (2015). Achievements and challenges in structural bioinformatics and computational biophysics. *Bioinformatics*, *31*(1), 146-150. doi:10.1093/bioinformatics/btu769
- Seemayer, S., Gruber, M., & Soding, J. (2014). CCMpred--fast and precise prediction of protein residue-residue contacts from correlated mutations. *Bioinformatics*, *30*(21), 3128-3130. doi:10.1093/bioinformatics/btu500
- Shelhamer, E., Long, J., & Darrell, T. (2017). Fully Convolutional Networks for Semantic Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, *39*(4), 640-651. doi:10.1109/tpami.2016.2572683
- Shen, M. Y., & Sali, A. (2006). Statistical potential for assessment and prediction of protein structures. *Protein Sci*, *15*(11), 2507-2524. doi:10.1110/ps.062416606
- Sims, G. E., & Kim, S.-H. (2006). A method for evaluating the structural quality of protein models by using higher-order phi-psi pairs scoring. *Proceedings of the National*

- Academy of Sciences of the United States of America*, 103(12), 4428-4432. doi:10.1073/pnas.0511333103
- Skwark, M. J., Abdel-Rehim, A., & Elofsson, A. (2013). PconsC: combination of direct information methods and alignments improves contact prediction. *Bioinformatics*, 29(14), 1815-1816.
- Soding, J. (2005). Protein homology detection by HMM-HMM comparison. *Bioinformatics*, 21(7), 951-960. doi:10.1093/bioinformatics/bti125
- Stahl, K., Schneider, M., & Brock, O. (2017). EPSILON-CP: using deep learning to combine information from multiple sources for protein contact prediction. *BMC bioinformatics*, 18(1), 303.
- Stein, A., & Kortemme, T. (2013). Improvements to robotics-inspired conformational sampling in rosetta. *PLoS One*, 8(5), e63090. doi:10.1371/journal.pone.0063090
- Tegge, A. N., Wang, Z., Eickholt, J., & Cheng, J. (2009). NNcon: improved protein contact map prediction using 2D-recursive neural networks. *Nucleic Acids Research*, 37(suppl_2), W515-W518.
- Uziela, K., Shu, N., Wallner, B., & Elofsson, A. (2016). ProQ3: Improved model quality assessments using Rosetta energy terms. *Sci Rep*, 6, 33509. doi:10.1038/srep33509
- Uziela, K., & Wallner, B. (2016). ProQ2: estimation of model accuracy implemented in Rosetta. *Bioinformatics*, 32(9), 1411-1413. doi:10.1093/bioinformatics/btv767
- Vendruscolo, M., Kussell, E., & Domany, E. (1997). Recovery of protein structure from contact maps. *Fold Des*, 2(5), 295-306. doi:10.1016/s1359-0278(97)00041-2
- Wang, G., & Dunbrack, R. L., Jr. (2003). PISCES: a protein sequence culling server. *Bioinformatics*, 19(12), 1589-1591. Retrieved from <https://www.ncbi.nlm.nih.gov/pubmed/12912846>
- Wang, J., Li, Z., & Shang, Y. (2017, 6-8 Nov. 2017). *New Deep Neural Networks for Protein Model Evaluation*. Paper presented at the 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI).
- Wang, S., Peng, J., Ma, J., & Xu, J. (2016). Protein Secondary Structure Prediction Using Deep Convolutional Neural Fields. *Sci Rep*, 6, 18962. doi:10.1038/srep18962
- Wang, S., Sun, S., Li, Z., Zhang, R., & Xu, J. (2016). Accurate De Novo Prediction of Protein Contact Map by Ultra-Deep Learning Model. doi:<http://dx.doi.org/10.1101/073239>
- Wang, Z., & Xu, J. (2013). Predicting protein contact map using evolutionary and physical constraints by integer programming. *Bioinformatics*, 29(13), i266-i273.
- Weigt, M., White, R. A., Szurmant, H., Hoch, J. A., & Hwa, T. (2009). Identification of direct residue contacts in protein-protein interaction by message passing. *Proc Natl Acad Sci U S A*, 106(1), 67-72. doi:10.1073/pnas.0805923106
- Wu, F. Y. (1982). The Potts model. *Reviews of Modern Physics*, 54(1), 235-268. doi:10.1103/RevModPhys.54.235
- Wu, S., & Zhang, Y. (2007). LOMETS: a local meta-threading-server for protein structure prediction. *Nucleic Acids Research*, 35(10), 3375-3382. doi:10.1093/nar/gkm251
- Wu, S., & Zhang, Y. (2008). A comprehensive assessment of sequence-based and template-based methods for protein contact prediction. *Bioinformatics*, 24(7), 924-931. doi:10.1093/bioinformatics/btn069

- Wu, Y., Lu, M., Chen, M., Li, J., & Ma, J. (2007). OPUS-Ca: a knowledge-based potential function requiring only Calpha positions. *Protein Sci*, *16*(7), 1449-1463. doi:10.1110/ps.072796107
- Xiang, Z., Soto, C. S., & Honig, B. (2002). Evaluating conformational free energies: the colony energy and its application to the problem of loop prediction. *Proc Natl Acad Sci U S A*, *99*(11), 7432-7437. doi:10.1073/pnas.102179699
- Xu, D., & Zhang, Y. (2012). Ab initio protein structure assembly using continuous structure fragments and optimized knowledge-based force field. *Proteins*, *80*(7), 1715-1735. doi:10.1002/prot.24065
- Yang, J., & Zhang, Y. (2015). Protein Structure and Function Prediction Using I-TASSER. *Current protocols in bioinformatics*, *52*, 5.8.1-5.8.15. doi:10.1002/0471250953.bi0508s52
- Yang, Y., & Zhou, Y. (2008). Specific interactions for ab initio folding of protein terminal regions with secondary structures. *Proteins*, *72*(2), 793-803. doi:10.1002/prot.21968
- Yeh, R. A., Chen, C., Lim, T.-Y., Hasegawa-Johnson, M., & Do, M. N. (2016). Semantic Image Inpainting with Perceptual and Contextual Losses. *CoRR*, *abs/1607.07539*.
- Yu, F., & Koltun, V. (2015). Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*.
- Zeiler, M. D., Krishnan, D., Taylor, G. W., & Fergus, R. (2010, 13-18 June 2010). *Deconvolutional networks*. Paper presented at the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.
- Zemla, A. (2003). LGA: A method for finding 3D similarities in protein structures. *Nucleic Acids Res*, *31*(13), 3370-3374. doi:10.1093/nar/gkg571
- Zhang, J., Wang, Q., Barz, B., He, Z., Kosztin, I., Shang, Y., & Xu, D. (2010). MUFOLD: A new solution for protein 3D structure prediction. *Proteins: Structure, Function, and Bioinformatics*, *78*(5), 1137-1152.
- Zhang, J., & Zhang, Y. (2010). A novel side-chain orientation dependent potential derived from random-walk reference state for protein fold selection and structure prediction. *PLoS One*, *5*(10), e15386. doi:10.1371/journal.pone.0015386
- Zheng, W., Li, Y., Zhang, C., Pearce, R., Mortuza, S. M., & Zhang, Y. (2019). Deep-learning contact-map guided protein structure prediction in CASP13. *Proteins*, *87*(12), 1149-1164. doi:10.1002/prot.25792
- Zhou, T., Shu, N., & Hovmoller, S. (2010). A novel method for accurate one-dimensional protein structure prediction based on fragment matching. *Bioinformatics*, *26*(4), 470-477. doi:10.1093/bioinformatics/btp679