

# UC Irvine

## UC Irvine Electronic Theses and Dissertations

### Title

Novel Machine Learning Approach for Protein Structure Prediction

### Permalink

<https://escholarship.org/uc/item/1336f301>

### Author

Nagata, Ken

### Publication Date

2014

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA,  
IRVINE

Novel Machine Learning Approach for Protein Structure Prediction

DISSERTATION

submitted in partial satisfaction of the requirements  
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Ken Nagata

Dissertation Committee:  
Professor Pierre Baldi, Chair  
Professor Richard Lathrop  
Professor Eric Mjolsness

2014

Portion of Chapter 2 © 2011 Wiley Periodicals, Inc.  
Portion of Chapter 2 © 2014 Oxford University Press  
Chapter 3 © 2012 Oxford University Press  
All other materials © 2014 Ken Nagata

# TABLE OF CONTENTS

	Page
<b>LIST OF FIGURES</b>	<b>iv</b>
<b>LIST OF TABLES</b>	<b>vi</b>
<b>ACKNOWLEDGMENTS</b>	<b>viii</b>
<b>CURRICULUM VITAE</b>	<b>ix</b>
<b>ABSTRACT OF THE DISSERTATION</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Protein Structure Prediction . . . . .	1
1.2 Contribution . . . . .	3
<b>2 Side Chain Prediction</b>	<b>4</b>
2.1 Introduction . . . . .	4
2.2 SIDEpro Algorithm . . . . .	8
2.2.1 ANN Architecture and Energy Function . . . . .	9
2.2.2 Training Data and Rotamer Targets . . . . .	12
2.2.3 Full Probabilistic Model . . . . .	14
2.2.4 Markov Chain Monte Carlo Optimization . . . . .	15
2.2.5 Fast Identification of Atom Contacts . . . . .	17
2.2.6 Elimination of Rotamers with Backbone Clashes . . . . .	20
2.2.7 Prediction of Rotamers . . . . .	20
2.2.8 Clash Reduction Procedure . . . . .	23
2.2.9 Incorporation of Fixed External Atoms into the Prediction Method . . . . .	24
2.3 SIDEpro for Non-Standard Amino Acids . . . . .	24
2.3.1 Training and testing datasets . . . . .	24
2.3.2 Building Rotamer Libraries for NSAs . . . . .	27
2.3.3 Training Energy and Prediction . . . . .	33
2.4 Results . . . . .	35
2.4.1 Evaluation Data and Protocol . . . . .	35
2.4.2 Accuracy . . . . .	36
2.4.3 CPU Time . . . . .	37
2.4.4 Clash Assessment . . . . .	40

2.4.5	Generic Energy vs Amino Acid Specific Energy . . . . .	41
2.4.6	Prediction of FPTMs . . . . .	42
2.4.7	Prediction of NSAs . . . . .	44
2.5	Discussion . . . . .	45
<b>3</b>	<b>Contact Map Prediction</b>	<b>58</b>
3.1	Introduction . . . . .	58
3.2	Materials and Methods . . . . .	60
3.2.1	Contact definition and evaluation criteria . . . . .	60
3.2.2	Training and test sets . . . . .	62
3.2.3	Coarse contact and orientation prediction (BRNN) . . . . .	63
3.2.4	Element alignment prediction (Energy) . . . . .	66
3.2.5	Residue-residue contact prediction (Deep NN) . . . . .	70
3.3	Results and Discussion . . . . .	78
3.3.1	Coarse contact and orientation prediction . . . . .	78
3.3.2	Element alignment prediction . . . . .	79
3.3.3	Residue-residue contact prediction: test set . . . . .	81
3.3.4	Residue-residue contact prediction: CASP test datasets . . . . .	84
3.4	Conclusion . . . . .	90
<b>4</b>	<b>Conclusion</b>	<b>91</b>
	<b>Bibliography</b>	<b>93</b>

# LIST OF FIGURES

	Page
2.1 Network Architecture . . . . .	11
2.2 Training pipeline. . . . .	31
2.3 Prediction pipeline for optimizing all the rotamer probabilities. Once the optimization is completed, final predictions are produced by first selecting the most likely rotamer and then going through a clash reduction algorithm. . . . .	32
2.4 CPU Times versus Protein Length for the 379 Protein SCWRL4 Dataset. The CPU time required by SIDEpro increases linearly with the number of residues, with a Pearson correlation of 0.97. For both SCWRL methods, the CPU time generally increases with the number of residues, but the relationships are less predictable. The Pearson correlation for SCWRL4-FRM is 0.53 and for SCWRL4-RRM it is 0.48. . . . .	39
3.1 Coarse contact map in (a) with its relation to the protein sequence for the CASP target T0604-D1 in (b). Blue, green, and red squares correspond to anti-parallel contact, parallel contact, and no-contact respectively. Coils, short helices ( $\geq 6$ ), and short strands ( $\leq 3$ ) are ignored and shown in black. . . . .	65
3.2 (a) Phase values (0-1) for pairs of residues in contacting strands. (b) Phase values (0-6) for pairs of residues in contacting helices. (c) Phase values for anti-parallel strands and corresponding energy terms $g_A(i, j, k)$ . . . . .	67
3.3 (a) Alignments and corresponding energies for two anti-parallel beta-strands of hypothetical length 3. (b) Alignment probabilities mapped to residue-residue contact probabilities for the two elements. . . . .	71
3.4 (a) The deep-NN architecture consists of a 3D stack of neural networks $NN_{ij}^k$ with identical architecture, but different weights. When $i$ and $y$ vary, the outputs of the $NN_{ij}^k$ correspond to the predicted contact map at level $k$ of the stack. A neural network $NN_{ij}^{k+1}$ purely spatial input features that depend only on $i$ and $j$ and are identical at all levels of the stack, and temporal input features associated with the contact probabilities predicted in the previous layer over a receptive field neighborhood of $ij$ . (b) Input feature vector of each $NN_{ij}^k$ . . . . .	72

3.5	Neighborhood of $(i, j)$ , or receptive field, of the level-dependent inputs. The temporal input feature vector (81 values) contains contact predictions computed in the previous layer of the stack. That is, for a pair of residues $(i, j)$ , the network-specific feature vector of $\text{NN}_{ijk+1}$ contains the contact predictions obtained by the networks $\text{NN}_{rsk}$ for all the pairs $(r, s)$ , where $r \in [i-l, \dots, i+l]$ , $s \in [j-l, \dots, j+l]$ and $l \geq 0$ is the “radius” of the neighborhood. We considered different topologies for the receptive pattern and different sizes for the radius parameter. In general, we found that larger radiuses provide better performance. On the other end, larger radiuses imply larger input and thus a longer training time and risk of overfitting. Our best results are obtained using a radius equal to 7 and the receptive field shown in Figure 3.5. The details of this receptive field are derived from the pattern of contacts in native contact maps between helical secondary structures. Since helical structures go around twice every seven residues or so, they tend to have contacts at $(i \pm 7, j \pm 7)$ when the residue pair $(i, j)$ is in contact. Furthermore, they also tend to have contacts at $(i \pm 7, j \pm 4)$ or $(i \pm 4, j \pm 7)$ because helical structures take a turn every 3-4 residues. . . . .	74
3.6	Contact proximity distribution for contacting residue pairs (continue line) and non-contacting residue pairs (dotted line), computed over long-range pairs of residues. . . . .	75
3.7	Cross validation performance of CMAPpro for the selection of the best network depth. The curve plots the <i>cross-validation accuracy - standard deviation over the ten validation sets</i> for the set of true number of long range contacts. The accuracy on the true number of long range contacts shows more variation than the accuracy on the L/5 long range contacts. The highest peak of accuracy is found at depth 71. . . . .	77
3.8	Accuracy (L/5 long range contacts) versus network depth for the set of test domains (All), the test domains of length between 50 and 100 residues (50-100), between 101 and 150 (>100-150), between 151 and 200 (> 150 – 200 and longer than 200 (>200)). The dotted vertical line indicates the actual depth of CMAPpro. . . . .	82
3.9	Native and predicted contact map for the T0604-D1 target from CASP9 set. The lower triangle shows the native contacts. The upper triangle shows contacts predicted by CMAPpro. The blue and red dots represent the correctly and incorrectly predicted contacts, respectively, among the L top-scored residue pairs. . . . .	85
3.10	Predicted contact map for the T0604-D1 target from CASP9 dataset. The lower triangle shows the predictions obtained with DNN and the upper triangle those obtained with NN+CA. The blue and red dots represent the correctly and incorrectly predicted contacts, respectively, among the L top-scored residue pairs. . . . .	86
3.11	Accuracy (L/5 long range contacts) versus network depth for the set of test domains (test), CASP8 domains (casp8) and CASP9 domains (casp9). The dotted vertical line indicates the actual depth of CMAPpro. . . . .	89

# LIST OF TABLES

	Page
2.1 The number of ANNs for each amino acid. Note that ALA and GLY are excluded because no side-chain search is needed for these residues. For CYS, we use two different ANNs for Sulphur-Sulphur interactions corresponding to S-S(CYS) and S-S(MET). . . . .	10
2.2 SIDEpro Training Set . . . . .	13
2.3 Frequent Post-Translational Modifications (FPTMs) . . . . .	27
2.4 Frequent Post-Translational Modifications (Continued) . . . . .	28
2.5 FPTM dataset . . . . .	47
2.6 FPTM dataset . . . . .	48
2.7 FPTM dataset . . . . .	49
2.8 FPTM dataset . . . . .	50
2.9 SCWRL4 Dataset Summary Results: 379 Proteins, 58229 Residues. . . . .	51
2.10 SCWRL4 Dataset Residue Specific Accuracy Results. . . . .	51
2.11 CASP9 Dataset Summary Results: 94 Proteins, 17885 Residues. . . . .	51
2.12 CASP9 Dataset Residue Specific Accuracy Results. . . . .	52
2.13 COMPLEXES Dataset Summary Results: 7 Protein Complexes with 91 Chains, 30734 Residues. . . . .	52
2.14 RIBOSOME Dataset Summary Results: 21 Protein Chains, 2001 Residues. . . . .	53
2.15 AA Specific Energy vs Generic Energy Tested on Standard Amino Acids . . . . .	53
2.16 PTM Specific vs Generic Energy for Frequent PTMs . . . . .	54
2.17 Accuracy for Frequent PTMs and their Precursor Amino Acid . . . . .	55
2.18 Accuracy of NSA Method on FPTM Set . . . . .	56
2.19 RMSD(Å) of NSA Method on FPTM Set . . . . .	56
2.20 Accuracy of NSA Method on NSA (non-FPTM) Test Set . . . . .	57
2.21 Statistical Significance (p-values) of SIDEpro Improvement with Respect to SCWRL4. Results significant at $p < 0.001$ are shown in bold. . . . .	57
3.1 Average performance for the coarse contact and orientation predictor. . . . .	78
3.2 Average accuracy on long-range contacts for the element alignment predictor. . . . .	80
3.3 Average accuracy and $X_d$ comparison on long-range contacts. . . . .	82
3.4 Average accuracy and $X_d$ on long-range contacts for CMAPpro. . . . .	82
3.5 Average $Acc$ and $X_d$ for seq. sep. $\geq 24$ on CASP8 set. . . . .	84
3.6 Average $Acc$ and $X_d$ for seq. sep. $\geq 24$ on CASP9 set. . . . .	87
3.7 CASP8 1-to-1 comparison of top ten predictors . . . . .	90



3.8	CASP9 1-to-1 comparison of top ten predictors . . . . .	90
-----	---	----

# ACKNOWLEDGMENTS

I would like to thank very sincerely Prof. Pierre Baldi for his guidance and assistance of my research. He provided me with informative advice and allowed me to work on projects that I am interested in. His prospective and ideas of different approaches widened my view on the way I should approach a research problem.

I would like to thank my committee, Prof. Eric Mjolsness and Prof. Richard Lathrop, for allowing me to be a part of their class as their Teacher's Assistant. The experience had helped me improve my presentation skills and gave me a better understanding of various topics discussed within the classes.

I also would like to thank Dr. Arlo Randall and Dr. Pietro Di Lena for their guidance and assistance throughout my PhD. career. I would not have been able to complete my publications without their help.

I also would like to thank all of my group members in Prof. Baldi's lab. I enjoyed the weekly discussions regarding individual projects as well as the occasional talks about personal lives.

Finally, I would like to thank my family for their love and support throughout my life. My older brother was always there to support me when I was in need for any kind of help, my parents had encouraged me to study abroad and obtain my PhD from a prestigious school, such as University of California - Irvine, and my wife Kazumi, has given me love and happiness during the 6 years that I have been attending this school.

Chapter 2 is mainly based on the following articles published in "Proteins" and "Bioinformatics".

- Nagata, K, Randall, A, Baldi, P (2012). SIDEpro: a novel machine learning approach for the fast and accurate prediction of side-chain conformations. *Proteins*, 80, 1:142-53.
- Nagata, K, Randall, A, Baldi, P (2014). Incorporating post-translational modifications and unnatural amino acids into high-throughput modeling of protein structures. *Bioinformatics*, 30, 12:1681-9.

Chapter 3 is mainly based on the article pulished in "Bioinformatics".

- Di Lena, P, Nagata, K, Baldi, P (2012). Deep architectures for protein contact map prediction. *Bioinformatics*, 28, 19:2449-57.

# CURRICULUM VITAE

Ken Nagata

## EDUCATION

<b>Doctor of Philosophy in Computer Science</b>	<b>2014</b>
<b>Master of Science in Computer Science</b>	<b>2010</b>
University of California, Irvine	<i>Irvine, California</i>
<b>Bachelor of Science in Electrical Engineering</b>	<b>2008</b>
Waseda University	<i>Tokyo, Japan</i>
<b>Exchange student</b>	<b>2007</b>
Arizona State University	<i>Tempe, Arizona</i>

## PUBLICATIONS

- Nagata, K, Randall, A, Baldi, P. Incorporating post-translational modifications and unnatural amino acids into high-throughput modeling of protein structures. *Bioinformatics*, 2014, 30, 12:1681-9.
- Di Lena, P., Nagata, K, Baldi, P., Deep Architectures for Protein Contact Map Prediction, *Bioinformatics*, 2012, 28, 2449-2457
- Di Lena, P., Nagata, K., Baldi, P., Deep Spatio-Temporal Architectures and Learning for Protein Structure Prediction. NIPS 2012 : Neural Information Processing Systems Conference. Accepted for presentation.
- Nagata, K., Randall, A. and Baldi, P., SIDEpro: A novel machine learning approach for the fast and accurate prediction of side-chain conformations. *Proteins*, 2012, 80, 142153.

# ABSTRACT OF THE DISSERTATION

Novel Machine Learning Approach for Protein Structure Prediction

By

Ken Nagata

Doctor of Philosophy in Computer Science

University of California, Irvine, 2014

Professor Pierre Baldi, Chair

The side-chain prediction and residue-residue contact prediction are sub-problems in the protein structure prediction. Both predictions are important for protein prediction and other applications.

We have developed a new algorithm, SIDEpro, for the side-chain prediction where an energy function for each rotamer in a structure is computed additively over pairs of contacting atoms. A family of 156 neural networks indexed by amino acids and contacting atom types is used to compute these rotamer energies as a function of atomic contact distances. Although direct energy targets are not available for training, the neural networks can still be optimized by converting the energies to probabilities and optimizing these probabilities using Markov Chain Monte Carlo methods. The resulting predictor SIDEpro makes predictions by initially setting the rotamer probabilities for each residue from a backbone-dependent rotamer library, then iteratively updating these probabilities using the trained neural networks. After convergences of the probabilities, the side-chains are set to the highest probability rotamer. Finally, a post processing clash reduction step is applied to the models. SIDEpro represents a significant improvement in speed and a modest, but statistically significant, improvement in accuracy when compared with the state-of-the-art for rapid side-chain prediction method SCWRL4 on the 379 protein test set of SCWRL4. Using the SCWRL4 test set, SIDEpro's

accuracy ( $\chi_1$  86.14%,  $\chi_{1+2}$  74.15%) is slightly better than SCWRL4-FRM ( $\chi_1$  85.43%,  $\chi_{1+2}$  73.47%) and it is 7.0 times faster. SIDEpro can also predict the side chains of proteins containing non-standard amino acids, including 15 of the most frequently observed PTMs in the Protein Data Bank and all types of phosphorylation. For PTMs, the  $\chi_1$  and  $\chi_{1+2}$  accuracies are comparable with those obtained for the precursor amino acid, and so are the RMSD values for the atoms shared with the precursor amino acid. In addition, SIDEpro can accommodate any PTM or unnatural amino acid, thus providing a flexible prediction system for high-throughput modeling of proteins beyond the standard amino acids.

We have also developed a novel machine learning approach for contact map prediction using three steps of increasing resolution. First, we use 2D recursive neural networks to predict coarse contacts and orientations between secondary structure elements. Second, we use an energy-based method to align secondary structure elements and predict contact probabilities between residues in contacting alpha-helices or strands. Third, we use a deep neural network architecture to organize and progressively refine the prediction of contacts, integrating information over both space and time. We train the architecture on a large set of non-redundant proteins and test it on a large set of non-homologous domains, as well as on the set of protein domains used for contact prediction in the two most recent CASP8 and CASP9 experiments. For long-range contacts, the accuracy of the new CMAPpro predictor is close to 30%, a significant increase over existing approaches.

Both SIDEpro and CMAPpro are part of the SCRATCH suite of predictors and available from: <http://scratch.proteomics.ics.uci.edu/>.

# Chapter 1

## Introduction

### 1.1 Protein Structure Prediction

Proteins are chains of amino acids and have a variety of functions in living organisms. They can bind and interact with other specific proteins or molecules like DNA. The structure defines molecule types to which the protein can bind or interact.

There are four levels of proteins structures: primary structure, secondary structure, tertiary structure, and quaternary structure. The primary structure of a protein is a sequence of amino acids. Since there are 20 types of amino acids, the primary structure is represented by a sequence of 20 letters. The secondary structure is a local structure of a protein. There are three common types of secondary structure: alpha-helix, beta-sheet, and random coil. Alpha-helix is a right-handed spiral structure and every four amino acids have a hydrogen bond, whereas the beta-sheet is a flat structure consists of several beta-strands. If the structures are neither an alpha-helix nor a beta-sheet, they are called a random coil. The tertiary structure is the 3D structure of a single protein. The quaternary structure is the structure of several proteins or protein complex.

Since the tertiary structure of a single protein or quaternary structure of protein complex can help the understanding or predicting its function, several experimental methods were invented to identify its structure. X-ray crystallography [39] and Nuclear Magnetic Resonance (NMR) spectroscopy [87, 7] are common methods. The structures determined by those methods are stored in The Protein Data Bank (PDB) [8], and more than 100,000 structures are currently available. The problem of experimental methods like X-ray crystallography and NMR spectroscopy is that it is very time consuming and expensive to get the structure of proteins.

Protein structure prediction is an unsolved problem in bioinformatics and computational biology. The objective of the problem is to predict the tertiary structure from the primary structure. Being able to accurately predict the tertiary structure, we will be able to solve the mystery of the protein folding process in living organisms. In addition, accurate tertiary structure prediction will speed up the experimental process because experimental methods will become unnecessary.

There are several sub-problems in protein structure prediction. The secondary structure prediction predicts the secondary structure from the primary sequence [68]. The contact map prediction predicts the pair of amino acid residues which are contacted in 3D space [17]. The side-chain prediction predicts the side-chain conformations from the backbone structure [45, 61]. I have worked on the contact-map prediction and the side-chain prediction. In Chapter 2, the details of new side-chain prediction algorithm, SIDEpro, will be introduced. In Chapter 3, the new contact-map algorithm using a neural networks will be introduced.

## 1.2 Contribution

My main contributions to the side chain, contact map prediction and other projects are as follows:

- Designed the SIDEpro algorithm for both training and prediction (Section 2.2)
- Extended the SIDEpro capability to predict side chain structures of non-standard amino acids (Section 2.3)
- Evaluated results for the SCWRL4 test set (Part of Section 2.4.2, and Section 2.4.3)
- Evaluated FPTMs and NSAs accuracy levels (Section 2.4.5-2.4.7)
- Created coarse contact maps, designed feature inputs for its prediction, and evaluated its method (Section 3.2.3 and Section 3.3.1)
- Created the algorithm for element alignment prediction and evaluated its method (Section 3.2.4 and Section 3.3.2)
- Designed the neighborhood pattern shown in Figure 3.5.
- Evaluated part of the residue-residue contact prediction (Section 3.3.3 and Section 3.3.4)
- Applied the deep architecture for 1D prediction
- Collaborated with Dr. Chang C. Liu on the HIV project to find the best sequence of a protein called 412d which has high affinity with a protein (gp120) produced by HIB virus. In this project, I predicted the protein structures for mutated sequences with SIDEpro, ran MD simulations, calculated binding energy, and applied the greedy search to find the sequences with lowest energy levels.
- Created input features for alpha-beta transmembrane classification



# Chapter 2

## Side Chain Prediction

### 2.1 Introduction

Protein structure prediction is a fundamental problem in computational molecular biology and predicting the side-chain conformations for a given fixed backbone is an important subproblem [73]. Side-chain prediction methods are also critical for protein engineering, protein design, protein-protein docking [33, 4]. In both protein structure prediction and flexible protein-protein docking, the side-chain conformation predictions are relatively time intensive, and can be the limiting factor in how much search space can be explored. Thus, improvements in both speed and accuracy are highly desirable.

As a result, several methods have been proposed to address the side-chain prediction problem. SCWRL is one of the best methods and is widely used because it is generally fast and accurate, in addition it is convenient to download and run locally [15, 45]. SCWRL finds a combination of rotamers which minimizes an energy function based on rotamer probabilities and physical/chemical energy terms [15, 45]. There are two types rotamer libraries, backbone-independent and backbone-dependent. Backbone-independent rotamer libraries

contain information on side-chain dihedral angles and rotamer populations. Backbone-dependent libraries contain the same information as a function of the backbone dihedral angles  $\phi$  and  $\psi$  [22, 23, 21, 20]. SCWRL, and most other successful prediction methods, utilize the probabilities from backbone-dependent rotamer libraries [15, 45] and some combination of physics-based energy terms based on electrostatics, hydrogen bonding, solvation free energy [25, 54], van der Waals forces or at least simple steric energy. In addition, many search methods have been applied to the side-chain prediction problem, such as dead-end elimination [32], Monte Carlo [51, 73, 25, 54], cyclical search [88, 89], self-consistent mean field optimization [44, 56], integer programming [43], and graph decomposition [91, 45]. These methods rely on reducing the search space to pre-defined discrete sets of conformers for each side-chain (rigid rotamers) in order to make rapid predictions [4].

While physics-based energy functions have been applied with some success to the side-chain prediction problem in combination with discretized rigid rotamer models, they are not ideally suited for handling the coarseness of such models. In particular, repulsive physics-based energy terms are quite sensitive to slight changes in atom-atom distances. As a result, in the most accurate discretized models, some of the side-chains interactions inevitably have much higher repulsive energies relative to the native structures. In contrast, knowledge-based energy functions extracted from large training sets, which have been widely used in the field of protein structure prediction (e.g. [78, 93]), can be more tolerant to discretization. Furthermore, these knowledge-based energy functions can potentially capture subtle effects not captured by the physics-based approximations, and they can be trained directly on the most accurate discretized models.

Motivated by these issues, here we develop a new kind of knowledge-based energy function which is designed specifically for the rigid rotamer search space, and incorporate it into a novel side-chain prediction method, SIDEpro, which surpasses SCWRL4 in speed and accuracy. SIDEpro uses a family of artificial neural networks (ANNs) that are trained to compute

an energy function based on atom-atom distances [6]. The structure models used to train the ANNs are modified versions of Protein Data Bank (PDB) structures [8], where each side-chain is independently set to the most accurate rigid rotamer. SIDEpro makes predictions by initially setting the probability of the rotamers for each residue using a backbone dependent rotamer library [22, 23, 21, 20]. Then it iteratively updates these rotamer probabilities using the ANNs energy until all the rotamer probabilities converge. Then, the side-chains are set to the rotamer with highest probability. Finally, a post-processing clash reduction procedure is applied to the models.

Post-translational modifications (PTMs) are critical to the function of many proteins in living systems and understanding their effects at the molecular level is important for both basic and applied research in biology and medicine. To further this understanding, open databases of curated PTM information have been published. For instance, Phospho.ELM [19] is a publicly available database dedicated to phosphorylation. The database provides the exact positions of experimentally determined phosphorylation sites as well as information on the specific kinases that produce the modifications. Other databases such as PhosphoSitePlus, HPRD, and PHOSIDA [36, 40, 30] include information on additional types of PTMs (e.g. ubiquitination, acetylation, methylation) but are still dominated by phosphorylation data. An automated curator of information on PTMs [41] found in Swiss-Prot [5] provides the following summary statistics: there are a total of 82,505 PTMs determined by experimental methods, with the following types having a frequency greater than 1%: phosphorylation - 70.9%; acetylation 8.2%; N-linked glycosylation 6.8%; amidation - 3.5%; hydroxylation - 2.0%; methylation - 1.9%; O-linked glycosylation - 1.4%; ubiquitylation - 1.1%; and pyrrolidone carboxylic acid - 1.0%.

In addition to methods for curating and organizing existing PTM data, there are also methods for predicting which sites are modified in sequences with unknown PTM status. These methods typically use supervised machine-learning, statistical, and motif based approaches

to predict sites of phosphorylation [11, 84], acetylation [49], glycosylation [34, 50, 38], sumoylation [92, 71], and less common types of PTMs as well [66]. Some of these methods predict both the specific phosphorylated sites and the specific kinases responsible for the phosphorylation [42, 12, 63].

In contrast with these approaches, the fundamental problem of predicting accurate 3D models of PTMs in proteins has been largely ignored. None of the widely used, or recently published side-chain prediction methods that are free for academic research [35, 54, 45, 94, 52, 61] are capable of incorporating PTMs or unnatural amino acids into their predictions. The widely used template-based modeling software Modeller [74] does allow for manual creation of custom residues; however, the process for doing so is somewhat cumbersome and not realistic for most Modeller users. For protein-peptide interface, incorporating unnatural amino acids into Rosetta[48] was successfully done [72].

We recognize the need for generating accurate models that incorporate PTMs; however, there are a number of practical challenges that have stymied progress in this area: (1) there is far less data in the Protein Data Bank (PDB) [8] for PTM residues than native residues for building rotamer libraries or developing statistical potentials; (2) while 1-character codes (e.g. A for Alanine) work well for efficiently defining protein sequences, it is unfeasible to use 1-character codes for all possible PTMs (there are over 100 PTMs documented in the literature); (3) some important modifications (e.g. O-linked glycosylation) correspond to broad classes of chemical structures rather than a unique chemical structure, and each of the possible molecules would need to be uniquely identified; and (4) modified residues are generally larger and contain more rotatable bonds than their natural counterparts.

Beyond the 20 standard amino acids and their PTMs, there are also other natural or synthetic amino acids that can be incorporated into proteins. Two additional natural amino acids, Selenocysteine (Sec,U) and Pyrrolysine (Pyl,O), are coded in some species by codons that are usually interpreted as stop codons. Pyrrolysine, for instance, is used by some methanogenic

archaea in enzymes used to produce methane. In addition, over 40 non-natural amino acids have been incorporated into proteins through synthetic biology projects, often by creating a unique codon (recoding) and a corresponding transfer-RNA, to explore protein structure and function and create novel proteins [90, 85]. A tool for modeling the side chains of these rare natural or non-natural amino acids would also be desirable.

Thus, despite the challenges described above, we have taken on the problem of rapidly generating reasonably accurate 3D side chain models of proteins that incorporate amino acids beyond the standard 20 amino acids. In the remainder of this article the term “Non-Standard Amino Acid” (NSA) is used to refer to any amino acid other than the 20 standard amino acids. This includes PTMs, rare natural amino acids, and unnatural amino acids.

In this chapter, we introduce the SIDEpro algorithm for natural amino acids in the section 2.2. Then, the training and prediction methods for non-standard amino acids will be shown in the section 2.3. The section 2.4 shows 1) comparative evaluation of SIDEpro and SCWRL4 for natural amino acids, and 2) evaluation for Frequent PTMs and other non-standard amino acids.

## 2.2 SIDEpro Algorithm

Here we present the details of the SIDEpro method in the following subsections: (1) the ANN architecture and energy function; (2) the training data and rotamer targets; (3) the full probabilistic model; (4) the Markov Chain Monte Carlo optimization procedure; (5) the fast identification of atom contacts; (6) the elimination of rotamers with backbone clashes; (7) the prediction of rotamers; (8) the clash reduction procedure; and (9) the incorporation of fixed external atoms into the prediction method. A pseudocode outline of the prediction steps is also presented in Algorithm 1.

### 2.2.1 ANN Architecture and Energy Function

At its core, SIDEpro utilizes a family of neural networks to compute an energy function. This is a computational energy function and is not necessarily meant to represent a physical quantity. Each neural network is specialized to compute a particular term of the total energy associated with a particular amino acid and the distance between a particular pair of atom types. Thus, for instance, there is a neural network for Cysteine specialized for Sulphur-Carbon distances. While we did experiment with various ANN architectures, here we report the results obtained by using a simple three layer perceptron ANN architecture, for each member of the family (Figure 2.1). Each ANN receives a single external input, corresponding to a distance  $d$  between a pair of atoms of the given type, and produces a single output corresponding to the “energy” of the pair. The hidden layers typically have 15-20 units. The family comprises 156 different ANNs (Table 1), indexed by three variables corresponding to: (1) the amino acid type; (2) the atom type of the first atom in the pair, restricted to side chain atoms (no backbone atoms); and (3) the atom type of the second atom in the pair, including backbone atoms. The set of atom types is {N,C,O,S,H}. When the distance between two atoms is large enough they are not considered to be interacting in our model, thus we set the energy to zero when the distance,  $d$ , is greater than 7 Å. Thus the output  $e$  of a single network can be expressed as (Figure 2.1):

$$e = f(d; \mathbf{w}) = \begin{cases} \sum_{h=1}^H w_{4h} \sigma(w_{2h}d + w_{1h}) + w_{31}, & d < 7 \\ 0, & \text{otherwise} \end{cases} \quad (2.1)$$

where

$\mathbf{w}$  : ANN weights

$\sigma$  : logistic sigmoid function ( $\sigma(x) = 1/(1 + e^{-x})$ )

$H$  : number of hidden states (15-20)

Note that a sigmoid function is not applied to the output layer because the output is not restricted to  $[0,1]$ .

Table 2.1: The number of ANNs for each amino acid. Note that ALA and GLY are excluded because no side-chain search is needed for these residues. For CYS, we use two different ANNs for Sulphur-Sulphur interactions corresponding to S-S(CYS) and S-S(MET).

Amino Acid	Number of ANNs
CYS	6
ASP	10
GLU	10
PHE	5
HIS	10
ILE	5
LYS	10
LEU	5
MET	10
ASN	15
PRO	5
GLN	15
ARG	10
SER	5
THR	10
VAL	5
TRP	10
TYR	10
Total	156

By adding the contributions of each neural network applied to each pair of interacting atoms it is possible to compute the total energy of each rotamer, or the total energy of a protein.

In particular, the energy of rotamer  $j$  for amino acid  $i$  is calculated as follows:

$$E_{ij} = \sum_{k=1}^{N_i} \sum_{l=1, l \neq i}^L \sum_{n=1}^{N_l} f_{a_i b_{ik} b_{ln}}(\|\mathbf{X}_{ijk} - \mathbf{X}_{lrln}\|; \mathbf{w}) \quad (2.2)$$

where

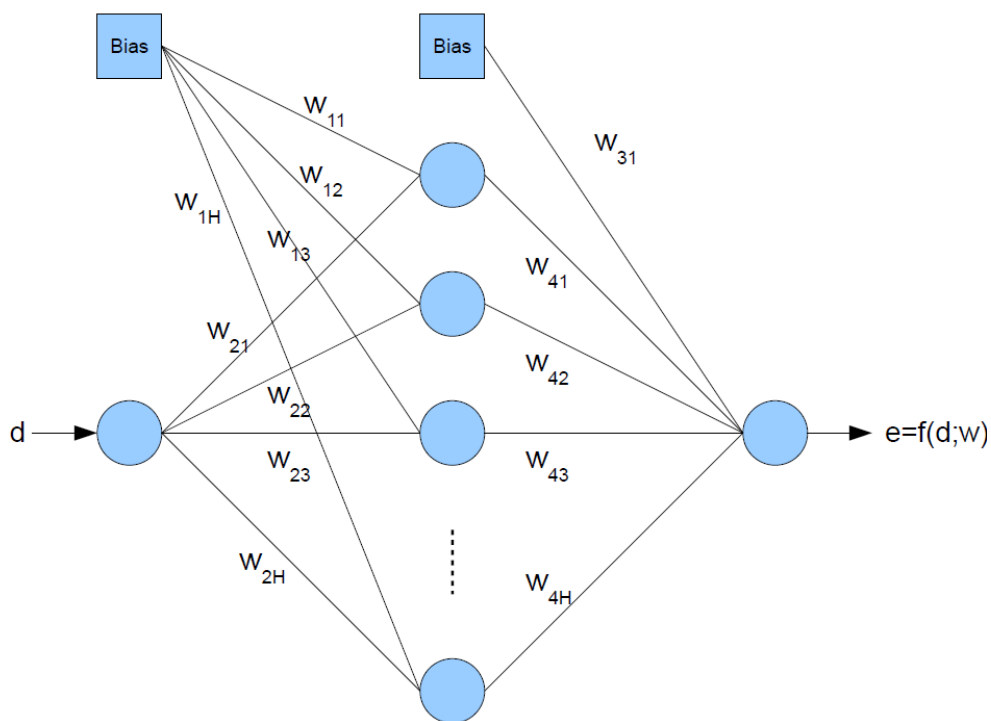


Figure 2.1: Network Architecture

$E_{ij}$  : training energy for rotamer  $j$  at residue  $i$

$N_i$  : the number of atoms of residue  $i$

$L$  : the length of the amino acid sequence

$\mathbf{X}_{ijk}$  : coordinate of  $k$ -th atom of rotamer  $j$  of residue  $i$

$a_i$  : amino acid type of residue  $i$

$b_{ik}$  : atom type of  $k$ -th atom of residue  $i$

$f_{a_i b_{ik} b_{ln}}(\cdot)$  : energy term computed by the ANN associated with the amino acid type of residue  $i$ , the atom type of the  $k$ -th atom of residue  $i$ , and the atom type of the  $n$ -th atom of residue  $l$

$r_i$  : rotamer target associated with residue  $i$  (see next section)



Here and in the rest of the paper we use bold-face fonts to denote vectors and contrast them with scalar variables.

A fundamental observation is that we do not have direct energy values that can be used to train the neural networks. However, we can derive rotamer targets from structures in the PDB [8]. As we shall see the energies computed by the ANNs can be converted into probabilities, and the ANNS can then be trained to maximize the probability of the target rotamers of the true structures. This optimization cannot be carried analytically but is implemented here using a Markov Chain Monte Carlo methods, essentially a variant of simulated annealing.

### 2.2.2 Training Data and Rotamer Targets

SIDEpro uses the publicly available backbone dependent rotamer library used by SCWRL3 [22][23][21][20][15] which was calculated from a set of 850 protein chains. The PISCES server was used to curate a diverse set of proteins to train the SIDEpro ANNs with the maximum mutual sequence identity set to 30%, maximum resolution of  $\leq 1.8 \text{ \AA}$ , and a maximum R-factor of 20%. 2,661 proteins resulted from this search and 300 were randomly selected. Next, the 48 proteins with at least 25% sequence identity with one or protein in the SCWRL4 dataset were removed from the training set leaving 252 proteins for training. Table 2.2 shows a list of PDB IDs in the SIDEpro training set.

Rather than using the raw PDB coordinates of these proteins, we set each side chain to the best matching rotamer in the corresponding rotamer library. Specifically, consider an amino acid  $i$  in a protein with  $R_i$  rotamers in the corresponding library. The library comes with a prior distribution  $p_{ij}$ ,  $j = 1, \dots, R_i$ . For each one of these rotamers, there is a vector of up to four angles  $\chi_{ij} = (\chi_{ij}^k)$ ,  $k = 1, \dots, 4$ , with four corresponding normal distributions with known mean and standard deviations [20], so that:  $P(\chi_{ij} | \mu_{ij}, \sigma_{ij}) = \prod_{k=1}^{k=4} P(\chi_{ij}^k | \mu_{ij}^k, \sigma_{ij}^k)$ .

Table 2.2: SIDEpro Training Set

PDB IDs
1A68, 1A7S, 1A8D, 1ABA, 1ADS, 1AKO, 1AL3, 1AOP, 1AQB, 1ARB, 1AYL, 1B6G, 1BFD, 1BFG, 1BG6, 1BJ7, 1BM8, 1BS9, 1BX7, 1C3D, 1CEM, 1CEX, 1CNZ, 1CSH, 1CTF, 1CTJ, 1CV8, 1DGW, 1DHN, 1DIN, 1DQG, 1E9G, 1ECA, 1EDG, 1EU1, 1EV5, 1EZM, 1FD3, 1FNA, 1FNC, 1GAI, 1GOF, 1GQ6, 1GVF, 1GYX, 1H2C, 1HKA, 1IFC, 1ISP, 1IXH, 1JER, 1JO0, 1K0M, 1K3X, 1KAP, 1KUH, 1LBU, 1LCL, 1LK9, 1LWB, 1M6K, 1MLA, 1MRP, 1N0W, 1N4W, 1NOX, 1NPS, 1O08, 1OAA, 1OCY, 1ODM, 1OFL, 1ONW, 1OPD, 1ORC, 1P4C, 1PDA, 1PZ4, 1QG8, 1QNF, 1RCQ, 1RHS, 1ROC, 1RTQ, 1RWH, 1RWR, 1RXY, 1RYC, 1S1D, 1SBP, 1SMB, 1T4B, 1T6E, 1TIF, 1TKJ, 1TT8, 1TYV, 1U53, 1UAE, 1UAS, 1UCR, 1US5, 1USH, 1UUJ, 1UWC, 1VBW, 1VHH, 1VLB, 1VQS, 1WC2, 1WKR, 1WV3, 1X1N, 1XLQ, 1XNB, 1XTE, 1XWW, 1Y6X, 1YBI, 1YCD, 1YD0, 1YE8, 1YGE, 1YKU, 1YNP, 1YWF, 1Z70, 1Z9N, 1ZIN, 1ZLD, 1ZND, 2A2K, 2ABS, 2AFW, 2AG4, 2AI4, 2AML, 2AYH, 2B0V, 2BBA, 2BF6, 2BJQ, 2BT9, 2BU3, 2BW4, 2CE2, 2D81, 2DSK, 2DT4, 2DTJ, 2DXQ, 2E2R, 2END, 2EPL, 2FCT, 2FR5, 2FUR, 2GS8, 2GX5, 2H1T, 2H30, 2H98, 2HA8, 2HBG, 2HJE, 2HW2, 2I02, 2I0O, 2I7D, 2IMZ, 2IUW, 2J3X, 2JLI, 2O7R, 2ODI, 2OGX, 2OIK, 2OIT, 2OML, 2OOC, 2OU6, 2P1M, 2P3P, 2PLR, 2PMQ, 2PR7, 2Q99, 2QJL, 2QS9, 2QSW, 2QZU, 2R31, 2RFP, 2RKQ, 2SAK, 2SNS, 2V25, 2V2G, 2VB1, 2VE8, 2VLG, 2VQ2, 2VYO, 2YVR, 2Z3V, 2Z51, 2Z72, 2ZK9, 2ZPT, 2ZSG, 2ZYJ, 3B7S, 3BCW, 3BLN, 3BMZ, 3BON, 3C1Q, 3C9Q, 3CHJ, 3CK1, 3CLA, 3CNH, 3CWN, 3CZX, 3D0J, 3D1P, 3D33, 3D3Z, 3DG9, 3DS2, 3DXY, 3EKG, 3ERP, 3F7L, 3FDH, 3FFR, 3FGH, 3FLA, 3FOJ, 3FPW, 3G5T, 3GA3, 3GFP, 3GJ0, 3GPG, 3GY9, 3H7C, 3HA9, 3HM4, 3I4Z, 3IAR, 3IRB

Thus for a given amino acid  $i$  in a training protein in the PDB with observed angles  $\chi_i^{PDB}$ , we can use Bayes theorem to compute a posterior probability for each rotamer in the corresponding family in the form:

$$P(r_{ij}|\chi_i^{PDB}) = \frac{P(\chi_i^{PDB}|r_{ij})p_{ij}}{P(\chi_i^{PDB})} \quad (2.3)$$

Thus we assign a rotamer  $r_i$  to this amino acid by maximizing the posterior:

$$r_i = \underset{j}{\operatorname{argmax}} P(\chi_i^{PDB}|\mu_{ij}, \sigma_{ij})p_{ij} \quad (2.4)$$

where  $\chi_i^{PDB}$  is the vector of  $\chi$  angles calculated from residue  $i$  in the native structure and

$p_{ij}$  is the probability of the  $j$ -th rotamer in the library for residue  $i$ .

### 2.2.3 Full Probabilistic Model

We can now proceed to use the energy computed by the ANNs to derive a full probabilistic model. The posterior probability of the training rotamer at residue  $i$  is the combination of the energy calculated by equation (2.2) when the weights of neural network  $\mathbf{w}$  are given, with the rotamer library probability and is defined by:

$$P(r_i|\mathbf{w}) = \frac{\exp(-K E_{ir_i})p_{ir_i}}{\sum_{j=1}^{R_i} \exp(-K E_{ij})p_{ij}} \quad (2.5)$$

where  $R_i$  is the number of rotamers for residue  $i$  and  $K$  is a constant ( $K = 0.1$ ). We make the following standard independence approximation

$$P(\mathbf{r}|\mathbf{w}) = \prod_i P(r_i|\mathbf{w}) \quad (2.6)$$

This equation can be treated as a likelihood function with the posterior distribution of  $\mathbf{w}$  given the data  $\mathbf{r}$  provided by Bayes theorem:

$$P(\mathbf{w}|\mathbf{r}) = \prod_i \frac{P(r_i|\mathbf{w})P(\mathbf{w})}{P(\mathbf{r})} \quad (2.7)$$

Here  $P(\mathbf{w})$  is the prior distribution on the weights  $\mathbf{w}$  of the ANNs and  $P(\mathbf{r})$  is a normalizing constant that does not depend on  $\mathbf{w}$  and can be ignored during the optimization process. Each ANN has its own prior distribution, which is modeled using a set of four independent zero-mean normal distributions in the form:

$$P(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{i=1}^4 \sqrt{\frac{\alpha_i}{2\pi}} \exp\left(-\frac{\alpha_i}{2} \sum_j w_{ij}^2\right) \quad (2.8)$$

where  $\boldsymbol{\alpha}$  is a vector of hyper parameters, and each component  $\alpha_i$  controls the distribution of the corresponding weights [62, 55].

To complete the description of the full hierarchical probabilistic model, the positive hyper-parameters  $\alpha_i$  can be assumed to be independent and follow a Gamma distribution

$$P(\boldsymbol{\alpha}) = \prod_{i=1}^4 \alpha_i^{k-1} \frac{\exp(-\alpha_i/\theta)}{\theta^k \Gamma(k)} \quad (2.9)$$

parameterized by  $k$  and  $\theta$ . After some experimentation, we fix the value of these parameters to  $k = 2$  and  $\theta = 0.5$ .

## 2.2.4 Markov Chain Monte Carlo Optimization

To optimize the ANN weights, we use a Markov Chain Monte Carlo method [62], essentially a form of simulated annealing described, for instance, in [59]. More precisely, the idea is to use Gibbs sampling to sample from the posterior of  $\alpha$ , and Metropolis sampling to sample from the distribution of  $\mathbf{w}$ . By assuming  $\mathbf{r}$  and  $\alpha$  are independent, the posterior of  $\mathbf{w}$  for a single ANN can be written as:

$$P(\mathbf{w}|\mathbf{r}, \boldsymbol{\alpha}) = \frac{P(\mathbf{w}, \mathbf{r}, \boldsymbol{\alpha})}{P(\mathbf{r}, \boldsymbol{\alpha})} = \frac{P(\mathbf{r}|\mathbf{w}, \boldsymbol{\alpha})P(\mathbf{w}|\boldsymbol{\alpha})P(\boldsymbol{\alpha})}{P(\mathbf{r})P(\boldsymbol{\alpha})} = \frac{P(\mathbf{r}|\mathbf{w})P(\mathbf{w}|\boldsymbol{\alpha})}{P(\mathbf{r})} \quad (2.10)$$

so that:

$$P(\mathbf{w}|\mathbf{r}, \boldsymbol{\alpha}) \propto \left( \prod_{i=1}^4 \sqrt{\frac{\alpha_i}{2\pi}} \right) \exp \left( - \sum_{i=1}^4 \frac{\alpha_i}{2} \sum_j w_{ij}^2 + \sum_i \log P(r_i|\mathbf{w}) \right) \quad (2.11)$$

Note that the last sum of Equation 2.11 ranges over all the amino acids in the training proteins where the corresponding ANN is used (e.g. for instance over all the cysteines in the set of training proteins). Similarly, for a single hyperparameter  $\alpha_i$  associated with a specific

ANN, we have

$$P(\alpha_i|\mathbf{r}, \mathbf{w}) = \frac{P(\alpha_i, \mathbf{r}, \mathbf{w})}{P(\mathbf{r}, \mathbf{w})} = \frac{P(\mathbf{r}|\mathbf{w}, \alpha_i)P(\mathbf{w}|\alpha_i)P(\alpha_i)}{P(\mathbf{r}, \mathbf{w})} = \frac{P(\mathbf{r}|\mathbf{w})P(\mathbf{w}|\alpha_i)P(\alpha_i)}{P(\mathbf{r}, \mathbf{w})} \quad (2.12)$$

so that:

$$P(\alpha_i|\mathbf{r}, \mathbf{w}) \propto \alpha_i^{k-1} \exp\left(-\alpha_i\left(\theta^{-1} + \frac{1}{2}\sum_j w_{ij}^2\right)\right) \propto \text{Gamma}\left(k, \left(\theta^{-1} + \frac{1}{2}\sum_j w_{ij}^2\right)^{-1}\right) \quad (2.13)$$

Thus a Markov Chain Monte Carlo method is applied to get samples of the posterior distribution  $P(\mathbf{w}, \boldsymbol{\alpha}|\mathbf{r})$  by iteratively using:

1. Gibbs sampling for  $\boldsymbol{\alpha}^{(j)}$
2. Metropolis sampling for  $\mathbf{w}^{(j)}$

Training proceeds by amino acid type, using all the ANNs associated with that particular amino acid type. For a given amino acid type and a given ANN, the learning algorithm alternates between Gibbs sampling and Metropolis sampling. Thus at the  $j$ -iteration, we first get samples of  $\boldsymbol{\alpha}^{(j+1)}$  by

$$\alpha_1^{(j+1)} \sim P(\alpha_1|\alpha_2^{(j)}, \alpha_3^{(j)}, \alpha_4^{(j)}, \mathbf{w}^{(j)}) \quad (2.14)$$

$$\alpha_2^{(j+1)} \sim P(\alpha_2|\alpha_1^{(j+1)}, \alpha_3^{(j)}, \alpha_4^{(j)}, \mathbf{w}^{(j)}) \quad (2.15)$$

$$\alpha_3^{(j+1)} \sim P(\alpha_3|\alpha_1^{(j+1)}, \alpha_2^{(j+1)}, \alpha_4^{(j)}, \mathbf{w}^{(j)}) \quad (2.16)$$

$$\alpha_4^{(j+1)} \sim P(\alpha_4|\alpha_1^{(j+1)}, \alpha_2^{(j+1)}, \alpha_3^{(j+1)}, \mathbf{w}^{(j)}) \quad (2.17)$$

Then we get a corresponding sample  $\mathbf{w}^{(j+1)}$  by:

1. get candidate sample using

$$\mathbf{w}^* \sim N(\mathbf{w}^j, \sigma^2) \quad (\text{with } \sigma^2 = 0.1) \quad (2.18)$$

2. accept sample  $\mathbf{w}^*$  as  $\mathbf{w}^{(j+1)}$  with probability  $\min \left[ \frac{P(\mathbf{w}^*|\mathbf{r}, \boldsymbol{\alpha}^{(j+1)})}{P(\mathbf{w}^{(j)}|\mathbf{r}, \boldsymbol{\alpha}^{(j+1)})}, 1 \right]$ , otherwise  $\mathbf{w}^{(j+1)} = \mathbf{w}^{(j)}$

The process is repeated for 500 iterations and the best observed weight  $\mathbf{w}^*$  is saved and used in prediction. The number of iterations was chosen empirically based on prediction accuracy results obtained on the training set.

$$\mathbf{w}^* = \operatorname{argmax}_{\mathbf{w}^{(j)}} P(\mathbf{w}^{(j)}|\mathbf{r}, \boldsymbol{\alpha}) \quad (2.19)$$

## 2.2.5 Fast Identification of Atom Contacts

In principle, the previous approach could require looking at all pairs of atoms in a protein in order to compute the corresponding energy term. However, most pairs of atoms are separated by a distance greater than the 7 Å cutoff and therefore are considered as non-interacting. Thus, in order to significantly reduce the calculation costs, and associated CPU time, SIDEpro identifies the spatial neighbors of each residue and then ignores all non-neighbors in the energy calculations. The corresponding pruning is implemented using simple bounding boxes: if two bounding boxes do not intersect, then the corresponding pair of atoms is separated by more than 7.0 Å and can be ignored from the energy calculation.

Neighbors are defined by checking for overlap between boxes in 3D space which are determined by subsets of residue atoms. For each residue  $i$ , a box in 3D space is calculated for the backbone atoms ( $3Dbox_{BB}(i)$ ), the side-chain atoms allowing all rotamers ( $3Dbox_{SC}(i)$ ), and for each rotamer  $j$  ( $3Dbox_{Rot}(i, j)$ ). Each box is defined as the space bound by six planes.

For  $3Dbox_{BB}(i)$  the planes are:

$$x = \min_{k \in backbone(i)} X_{i1k_x} - 3.5 \quad (2.20)$$

$$x = \max_{k \in backbone(i)} X_{i1k_x} + 3.5 \quad (2.21)$$

$$y = \min_{k \in backbone(i)} X_{i1k_y} - 3.5 \quad (2.22)$$

$$y = \max_{k \in backbone(i)} X_{i1k_y} + 3.5 \quad (2.23)$$

$$z = \min_{k \in backbone(i)} X_{i1k_z} - 3.5 \quad (2.24)$$

$$z = \max_{k \in backbone(i)} X_{i1k_z} + 3.5 \quad (2.25)$$

The planes which determine  $3Dbox_{SC}(i)$  are defined in the same manner, but the coordinates of all rotamers are searched:

$$x = \min_{j \in rotamers(i), k \in side-chain(i)} X_{ijk_x} - 3.5 \quad (2.26)$$

$$x = \max_{j \in rotamers(i), k \in side-chain(i)} X_{ijk_x} + 3.5 \quad (2.27)$$

$$y = \min_{j \in rotamers(i), k \in side-chain(i)} X_{ijk_y} - 3.5 \quad (2.28)$$

$$y = \max_{j \in rotamers(i), k \in side-chain(i)} X_{ijk_y} + 3.5 \quad (2.29)$$

$$z = \min_{j \in rotamers(i), k \in side-chain(i)} X_{ijk_z} - 3.5 \quad (2.30)$$

$$z = \max_{j \in rotamers(i), k \in side-chain(i)} X_{ijk_z} + 3.5 \quad (2.31)$$

The planes which determine  $3Dbox_{Rot}(i, j)$  are defined in the same manner, but using the

side-chain atoms of a specific rotamer:

$$x = \min_{k \in \text{side-chain}(i)} \mathbf{X}_{ijk_x} - 3.5 \quad (2.32)$$

$$x = \max_{k \in \text{side-chain}(i)} \mathbf{X}_{ijk_x} + 3.5 \quad (2.33)$$

$$y = \min_{k \in \text{side-chain}(i)} \mathbf{X}_{ijk_y} - 3.5 \quad (2.34)$$

$$y = \max_{k \in \text{side-chain}(i)} \mathbf{X}_{ijk_y} + 3.5 \quad (2.35)$$

$$z = \min_{k \in \text{side-chain}(i)} \mathbf{X}_{ijk_z} - 3.5 \quad (2.36)$$

$$z = \max_{k \in \text{side-chain}(i)} \mathbf{X}_{ijk_z} + 3.5 \quad (2.37)$$

Once the boxes are calculated, neighbor sets are defined for the backbone, side-chain, and each rotamer of each residue. The backbone neighbor set of residue  $i$ ,  $Nbr_{BB}(i)$ , consists of all residues  $l$  such that  $3Dbox_{BB}(l)$  intersects  $3Dbox_{SC}(i)$ . The side-chain neighbor set,  $Nbr_{SC}(i)$ , consists of all residues  $l$  such that  $3Dbox_{SC}(l)$  intersects  $3Dbox_{SC}(i)$ . The rotamer neighbor set,  $Nbr_{Rot}(i, j)$ , consists of all rotamers  $l, m$  such that  $3Dbox_{Rot}(l, m)$  intersects  $3Dbox_{Rot}(i, j)$ . Note, however, that only the rotamers of residue-level neighbors need to be checked. This is a simple version of the approach used in SCWRL4 to construct bounding boxes for checking for clashes [45], but it is used here efficiently to avoid calculating distances for the vast majority of residue and atom pairs.



## 2.2.6 Elimination of Rotamers with Backbone Clashes

Once the residue neighbors are defined, the next step is to eliminate any rotamers which clash with the protein backbone. We define a rotamer’s backbone clashing energy as:

$$E_{ij}^{VDW} = \sum_{k \in \text{side-chain}(i)} \sum_{l \in \text{Nbr}_{BB}(i)} \sum_{n \in \text{backbone}(l)} \begin{cases} (vdw_{ik} + vdw_{ln} - \|\mathbf{X}_{ijk} - \mathbf{X}_{ln}\|)^2, & \frac{\|\mathbf{X}_{ijk} - \mathbf{X}_{ln}\|}{vdw_{ik} + vdw_{ln}} < 0.67 \\ 0, & \text{otherwise} \end{cases} \quad (2.38)$$

where  $vdw_{ik}$  is the van der Waals radii of atom  $k$  of residue  $i$ . If  $E_i^{VDW} > 1.0$  the rotamer is considered to be clashing with the backbone and it is excluded from the search. An exception is made if all of the rotamers are clashing according to this criteria, in which case the least clashing rotamer (with lowest clashing energy) is used and all others are excluded. This approach reduces the search space significantly: on the SCWRL4 dataset approximately 20% of the rotamers are excluded from the search, resulting in a reduction of approximately 10% in average CPU time.

## 2.2.7 Prediction of Rotamers

After the rotamers which result in backbone clashes are excluded, the next step is to calculate the interaction energies between each rotamer of each residue and the entire backbone ( $E^{BB}$ ) and the side-chains of the previously defined neighboring residues ( $E^{SC}$ ). These interaction

energies are calculated as:

$$E_{ij}^{BB} = \sum_{k \in \text{side-chain}(i)} \sum_{l \in \text{Nbr}_{BB}(i)} \sum_{n \in \text{backbone}(l)} f_{a_i b_{ik} b_{ln}} (\|\mathbf{X}_{ijk} - \mathbf{X}_{l1n}\|; \mathbf{w}^*) \quad (2.39)$$

$$E_{ijlm}^{SC} = \sum_{k \in \text{side-chain}(i)} \sum_{n \in \text{side-chain}(l)} f_{a_i b_{ik} b_{ln}} (\|\mathbf{X}_{ijk} - \mathbf{X}_{lmn}\|; \mathbf{w}^*) \quad (2.40)$$

where  $i$  and  $l$  represent amino acids, and  $j$  and  $m$  represent rotamers in the corresponding libraries. This decomposition is computationally efficient because the backbone is fixed and thus the backbone contribution  $E^{BB}$  is constant.

The probability estimates for each rotamer are initially set to the values from the rotamer library. Then, the structure-dependent energy values,  $E^{BB}$  and  $E^{SC}$ , are combined with the rotamer library probabilities to calculate the posterior probabilities for each rotamer. The predicted energy of the individual rotamers ( $E_{ij}^P$ ) and the corresponding probability estimates ( $q_{ij}$ ) are updated iteratively. Thus we first initialize the probability  $q_{ij}$  using the rotamer library default probabilities  $q_{ij} = p_{ij}$  and we then iterate the following two update equations:

$$E_{ij}^P = E_{ij}^{BB} + \sum_{l, m \in \text{Nbr}_{Rot}(i, j)} q_{lm} E_{ijlm}^{SC} \quad (2.41)$$

$$q_{ij} = \frac{\exp(-K E_{ij}^P) p_{ij}}{\sum_{k=1}^{R_i} \exp(-K E_{ik}^P) p_{ik}} \quad (2.42)$$

SIDEpro uses only 6 iterations of this procedure, which in general is sufficient for the values to converge and yield a predicted rotamer for each residue  $i$  by setting:  $\hat{r}_i = \underset{j}{\operatorname{argmax}} q_{ij}$ .

---

**Algorithm 1** SIDEpro Pseudocode.

---

```
Calculate boxes:  $3D\mathit{box}_{BB}$ ,  $3D\mathit{box}_{SC}$ , and  $3D\mathit{box}_{Rot}$ 
Calculate neighbor sets:  $Nbr_{BB}$ ,  $Nbr_{SC}$ , and  $Nbr_{Rot}$ 
//set  $q_{ij}$  to be  $p_{ij}$  and remove clashing rotamers
for all residue  $i$  do
  for all rotamers at residue  $j$  do
     $q_{ij}=p_{ij}$ 
     $E_{vdw}(i, j) = \text{calculate equation (2.38)}$ 
    if  $E_{vdw}(i, j) > 1.0$  then
      remove rotamer  $j$ 
    end if
  end for
end for
//calculate  $E_{BB}$ 
for all residue  $i$  do
  for all rotamers  $j$  at residue  $i$  do
     $E_{BB}(i, j) = \text{calculate equation (2.39)}$ 
  end for
end for
//calculate  $E_{SC}$ 
for all residue  $i$  do
  for all rotamers  $j$  at residue  $i$  do
    for all  $l, m \in Nbr_{Rot}(i, j)$  do
       $E_{SC}(i, j, l, m) = \text{calculate equation (2.40)}$ 
    end for
  end for
end for
// update energies and probabilities until  $\mathbf{r}$  converges
while any  $q_{ij}$  does not converge do
  for all residue  $i$  do
    //update energy of rotamers at residue  $i$ 
    for all rotamers  $j$  at residue  $i$  do
       $e_{ij} = \text{calculate equation (2.41)}$ 
    end for
    //update probability of rotamers at residue  $i$ 
    for all rotamers  $j$  at residue  $i$  do
       $q_{ij} = \text{calculate equation (2.42)}$ 
    end for
  end for
end while
for all residue  $i$  do
   $\hat{r}_i = \underset{j}{\operatorname{argmax}} q_{ij}$ 
end for
```

---

---

**Algorithm 2** SIDEpro Pseudocode (continued).

---

where

$q_{ij}$  : updated probability of rotamer  $j$  at residue  $i$

$p_{ij}$  : initial probability of rotamer  $j$  at residue  $i$  (from library)

$\mathbf{r}$  : current set of rotamers  $i$

$\hat{r}_i$  : predicted rotamer for residue  $i$

---

## 2.2.8 Clash Reduction Procedure

The initial model produced by SIDEpro uses the means of the rotamer angles, thus it can be considered to be a rigid rotamer model. When a rigid rotamer model is used, some clashes are almost always observed in a compact protein model. In fact, the structures used for training contain some clashes because the rotamers were set to the mean of the closest rotamer in the library instead of using the raw coordinates. Since the SIDEpro energy function is a computational energy function learnt from data that contains some clashes, the initial model produced by SIDEpro may contain a certain number of steric clashes. To resolve these clashes we use a post-processing method which incorporates some flexibility into the rotamers.

The clashes are minimized using the following protocol. First, each residue is checked for clashes and for those with one or more clashing atoms, the  $\chi$  angles are updated to the mean ( $\mu$ )  $\pm$  the standard deviation ( $\sigma$ ) multiplied by  $m$  (initially set to 0.5), then the distances between atoms are recalculated and the number of clashes are recounted. Next, the  $\chi$  angles are set such that the number of clashes is minimized among the combination of  $\mu \pm \sigma$ . This operation is repeated until the clash counts converge. When the clash counts converge, the value of  $m$  is increased to 1.0 and the process is repeated. When the clash counts converge, the value of  $m$  is again increased to 1.5 and the process is repeated for the final time. This method reduces the number of clashes without a noticeable effect on accuracy.

## 2.2.9 Incorporation of Fixed External Atoms into the Prediction Method

SIDEpro can optionally incorporate fixed atoms from external molecules (other proteins, RNA, DNA, ligands, etc.) into the prediction process. These external fixed atoms are handled in a manner similar to backbone atoms. SIDEpro builds boxes and neighbors for them, and calculates the energy  $E_{BB}$  as if the fixed atom were backbone atoms. Since the SIDEpro ANNs are trained only on C, O, N, H and S atoms, any other atoms types are simply treated as C. In addition to handling external molecules, a subset of side-chains in the protein being predicted can be treated as fixed. These are handled by ignoring the rotamers at the fixed positions and using the input coordinates for all calculations.

## 2.3 SIDEpro for Non-Standard Amino Acids

### 2.3.1 Training and testing datasets

Since we use machine learning methods to predict the side chain conformations of NSAs, we first describe our curated datasets. We distinguish the 15 more frequent PTMs from all the other NSAs, since there is far more data available for them in the PDB.

#### Non Standard Amino Acid Dataset

The Protein Data Bank assigns a 3-letter identifier to unique chemical structures. The system is used for standard amino acids as well as other chemical structures (e.g. ligands, non-standard amino acids) that have coordinates in PDB files. To curate a set of NSAs observed in protein structures, we started from a set of 1449 chemical structures identified as

“non-standard polymeric components” by the PDB. From this starting set, we first removed molecules that were not amino acids, leaving 614 amino acids after this step. Then we downloaded all of the PDB structure files that contained one or more of these 614 identifiers yielding a total of 12,294 PDB files. Next, we checked the integrity of the peptide backbone for each potential NSA. If either peptide bond distance was greater than 1.5 Å, the NSA was excluded from the dataset, leaving 603 NSAs after this step. Next, we excluded any NSA that did not have at least one standard amino acid adjacent to it in the peptide chain. After this step, 549 distinct NSAs contained in 12,045 PDB files remained. The reason for this step was to exclude NSAs observed only in short peptides composed exclusively of NSAs, that are never observed integrated into proteins. Then, we excluded NSAs which have no carbon  $\gamma$ , or multiple carbon  $\gamma$ s, because only amino acids with a single  $\chi_1$  angle are considered for the prediction stage. After this step, 459 NSAs contained in 11,543 PDB files remained. Next, we excluded pdbs which have NSAs with high B-factors ( $>40$ ) because those side chain conformations are doubtful. Finally, we removed redundancy at the protein sequence level using a sequence similarity threshold of 30% and set aside the data corresponding to the 15 most frequent PTMs (see next section on FPTMs). The final NSA (non-FPTM) dataset consists of 316 unique NSAs contained in 1,308 PDBs files. The NSA (non-FPTM) data set is used exclusively for estimating the generalization accuracy of SIDEpro (see below).

### **Frequent PTMs Dataset**

Our main criteria for categorizing an NSA as a PTM was that a substructure of the NSA must be one of the standard 20 amino acids. We sought to discover the set of PTMs with sufficient instances in the PDB to allow for training and creating rotamer libraries. For this purpose we set a threshold of at least 50 instances. We sorted the curated NSA dataset by the total number of times the NSA is observed in the PDB. Multiple occurrences in the same PDB file were counted as unique occurrences. After ordering the dataset we observed

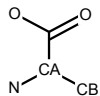
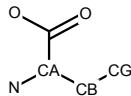
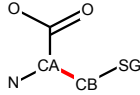
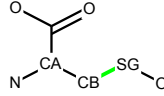
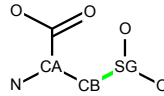

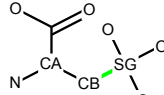
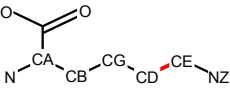
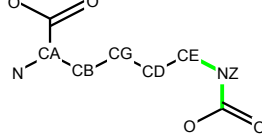
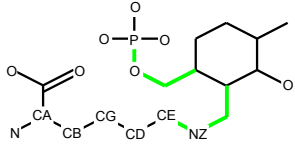
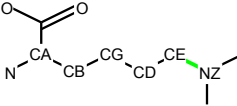
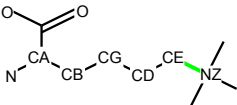
that there were 15 NSAs with 50 or more occurrences, and all of them were PTMs according to our definition. Tables 2.3 and 2.4 show the chemical structures of the PTMs and their precursor standard amino acids (e.g. tyrosine is the precursor of phosphotyrosine) using the PDB atom naming scheme to label individual atoms.

Selenomethionine (MSE) was associated with a particularly large number of PDB files, and thus we selected 500 of them at random.

Finally, for each PTM, we split the corresponding files were split into five folds to use cross validations. The total number of PDB files in the FPTM set is 1168.

Table 2.5, 2.6, 2.7 and 2.8 show PDB 3-letter code and the corresponding list of PDB file names in the FPTM set.

Table 2.3: Frequent Post-Translational Modifications (FPTMs)

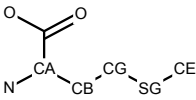
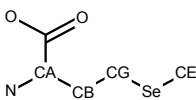
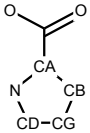
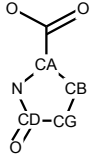
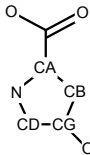
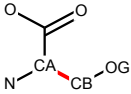
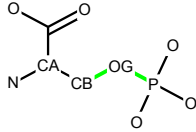
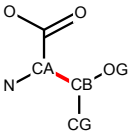
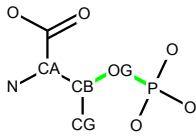
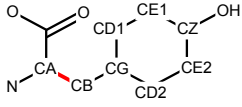
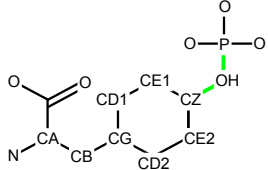
Precursor		PTM		
AA	Structure	ID	Name	Structure
ALA		ABA	ALPHA-AMINOBUTYRIC ACID	
CYS		CSO	S-HYDROXYCYSTEINE	
		CSD	3-SULFINOALANINE	
		CME	S,S-(2-HYDROXYETHYL) THIOCYSTEINE	
		OCS	CYSTEINESULFONIC ACID	
LYS		KCX	LYSINE NZ-CARBOXYLIC ACID	
		LLP	2-LYSINE(3-HYDROXY-2-METHYL-5-PHOSPHONOXYMETHYL-PYRIDIN-4-YLMETHANE)	
		MLY	N-DIMETHYL-LYSINE	
		M3L	N-TRIMETHYLLYSINE	

### 2.3.2 Building Rotamer Libraries for NSAs

A *fixed rotamer* is defined by a specific set of  $\chi$  angles whose values are typically equal to the mean of the values observed in a set of corresponding side-chain conformations that cluster in 3D space. A *flexible* rotamer is defined by both the means and variances of each one of



Table 2.4: Frequent Post-Translational Modifications (Continued)

Precursor		PTM		
AA	Structure	ID	Name	Structure
MET		MSE	SELENOMETHIONINE	
PRO		PCA	PYROGLUTAMIC ACID	
		HYP	4-HYDROXYPROLINE	
SER		SEP	PHOSPHOSERINE	
THR		TPO	PHOSPHOTHREONINE	
TYR		PTR	O-PHOSPHOTYROSINE	

its  $\chi$  angles. Both types of rotamers are widely used in side-chain conformation prediction, with rigid rotamer libraries [75, 9] generally leading to faster, but slightly less accurate, algorithms than flexible rotamer libraries [45, 53, 61]. While several rotamer libraries have been published for natural amino acids, a few libraries for NSAs have been published [29, 72].

### Flexible Rotamer Library for FPTMs

For 14 of the 15 PTMs in this study, the atoms of the precursor amino acid that is being modified are a subset of the atoms in the modified residue. The exception is Selenomethio-

nine. Of the 14 FPTMs where the precursor atoms are a subset, 12 introduce new rotatable bonds (i.e. additional  $\chi$  angles) that must be dealt with. The two FPTMs with proline as the precursor are the exceptions. For instance, Serine (SER) has only one  $\chi$  angle, whereas phosphorylated Serine (SEP) has three  $\chi$  angles.

In Tables 2.3 and 2.4, when FPTMs have additional  $\chi$  angles, the last  $\chi$  angle of the precursor amino acid is highlighted in red, and the additional  $\chi$  angles in the FPTM are highlighted in green. For instance, in Table 2.4 the last  $\chi$  angle of Serine, corresponding to the CA-CB bond, is highlighted in red. The two additional  $\chi$  angles, corresponding to the CB-OG and OG-P bonds in Phosphoserine, are highlighted in green. Note that for Phospho-Tyrosine (PTR), with Tyrosine (TYR) as the precursor, the first  $\chi$  angle is treated as the last  $\chi$  angle because the second (and final)  $\chi$  angle corresponding to the CB-CG bond is non-rotameric [77].

The  $\chi$  angles present in the precursor will be denoted by  $\chi_p$  and those that are additional in the modified residue by  $\chi_a$ . To model the  $\chi$  angles in FPTM residues that are present in the precursor residues ( $\chi_p$ ), a standard native amino acid rotamer library was used without modification [77]. The additional  $\chi$  angles in  $\chi_a$  were handled with a new customized method designed to accommodate cases where only few training instances are available, relative to the case of natural amino acids. For each FPTM type, except LLP and CME, we placed each  $i$ -th  $\chi$  angle ( $\chi_{ai}$ ) in  $\chi_a$  into one of three angle bins:  $(0^\circ, 120^\circ)$ ,  $(120^\circ, 240^\circ)$ , and  $(240^\circ, 360^\circ)$ . We calculated the corresponding means  $\mu_{ai}^{r_{ai}}$  and standard deviations  $\sigma_{ai}^{r_{ai}}$  where  $r_{ai}$  is a rotamer type for  $\chi_{ai}$ . By assuming that each  $\chi$  angle is independent,  $\chi_a$  can be assigned to a maximum of  $R_a = 3^{|\chi_a|}$  rotamers (rotamers with zero counts are eliminated).

For symmetric bonds (O-P bonds in LLP, SEP, TPO and PTR; CB-SG bond in OCS; NZ-C bond in KCX; and CE-NZ bond in M3L), since their  $\chi$  angles are almost constant, we set their mean  $\chi$  angle to  $180^\circ$  in the rotamer library. The  $\chi$  angles for PCA are also constant, and thus we set  $\chi_1 = 0^\circ$ ,  $\chi_2 = 0^\circ$ , and  $\chi_3 = 180^\circ$  for PCA. In all these cases, we set the

standard deviations to a small default value equal to  $10^\circ$ .

For LLP and CME, since they have many additional  $\chi$  angles and more possible rotamers, we found that the prediction accuracy is lower comparing to other FPTMs when using the library defined above. Because of this, we decreased the number of possible rotamers by decreasing the size of the bins. For LLP, the bins are:  $(0^\circ, 120^\circ)$  and  $(120^\circ, 360^\circ)$  for  $\chi_{a1}$ ;  $(0^\circ, 240^\circ)$  and  $(240^\circ, 360^\circ)$  for  $\chi_{a2}$  and  $\chi_{a4}$ ;  $(0^\circ, 180^\circ)$  and  $(180^\circ, 360^\circ)$  for  $\chi_{a3}$ ; and a single bin  $(0^\circ, 360^\circ)$  for  $\chi_{a5}$ . For CME, the bins are:  $(0^\circ, 180^\circ)$  and  $(180^\circ, 360^\circ)$  for  $\chi_{a1}$  and  $\chi_{a2}$ ; and we treated  $\chi_{a3}$  and  $\chi_{a4}$  as fixed bonds with values  $180^\circ$  and  $300^\circ$ . These bins were determined from the empirical distribution of  $\chi_{\mathbf{a}}$ .

We assume that  $\chi_{\mathbf{a}}$  is dependent on the last  $\chi$  angle in  $\chi_{\mathbf{p}}$ , marked in red in Tables 2.3 and 2.4, and referred to as  $\chi_{p,last}$ . This angle ( $\chi_{p,last}$ ) is associated with one of three bins of equal size  $120^\circ$  as above. For each rotamer of  $\chi_{p,last}$ , we calculated the rotamer probabilities  $p(\mathbf{r}_{\mathbf{a}}|r_{p,last})$ , where  $\mathbf{r}_{\mathbf{a}}$  is the rotamer types for the additional  $\chi$  angles, and  $r_{p,last}$  is the rotamer type for  $\chi_{p,last}$ . Since there are  $R_a$  rotamers for the additional  $\chi$  angles in  $\chi_a$  and  $R_p$  rotamers for the precursor residue, the total number of rotamers for a FPTM is  $R_a \times R_p$ , and the probability of combined rotamer  $(\mathbf{r}_{\mathbf{p}}, \mathbf{r}_{\mathbf{a}})$  is  $p_{\mathbf{p}}^{\mathbf{r}_{\mathbf{p}}} \times p(\mathbf{r}_{\mathbf{a}}|r_{p,last})$  normalized by the sum of all  $R_a \times R_p$  probabilities where  $\mathbf{r}_{\mathbf{p}}$  is a rotamer for the precursor residue.

### **Restricted Flexible Rotamer Library for NSAs(non-FPTM)**

Our approach to the generic prediction of NSAs, which do not correspond to FPTMs, treats only the first  $\chi_1$  (usually CA-CB) as rotatable, and considers the rest of the NSA structure as fixed. We built a general backbone independent flexible rotamer library for the  $\chi_1$  angle using the original SIDEpro training data set [61] (listed in Table 2.2). First, the  $\chi_1$  angles for all natural amino acids (except Alanine and Glycine which have no  $\chi_1$  angle) in the training set combined (not type specific) were calculated and placed into one of three bins:  $(0^\circ, 120^\circ)$ ,

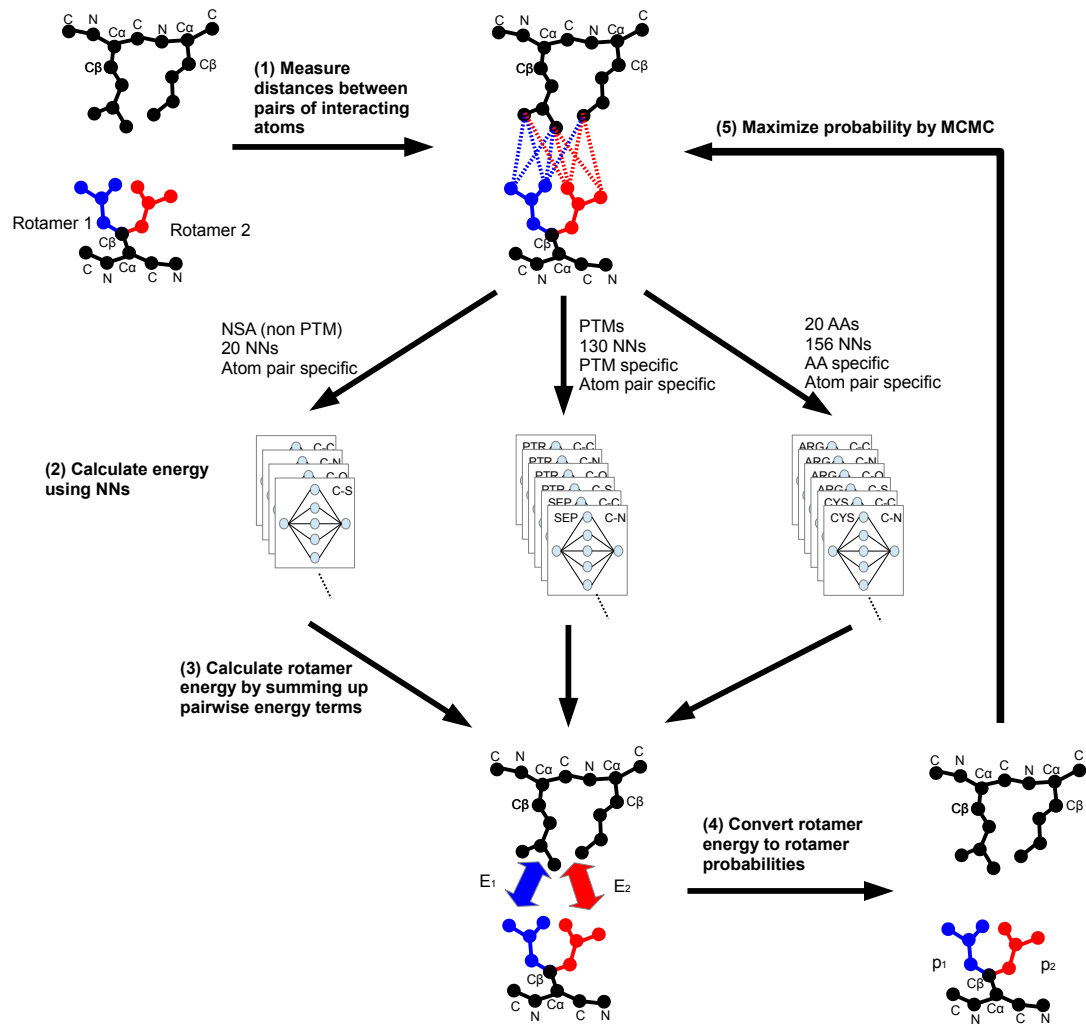


Figure 2.2: Training pipeline.

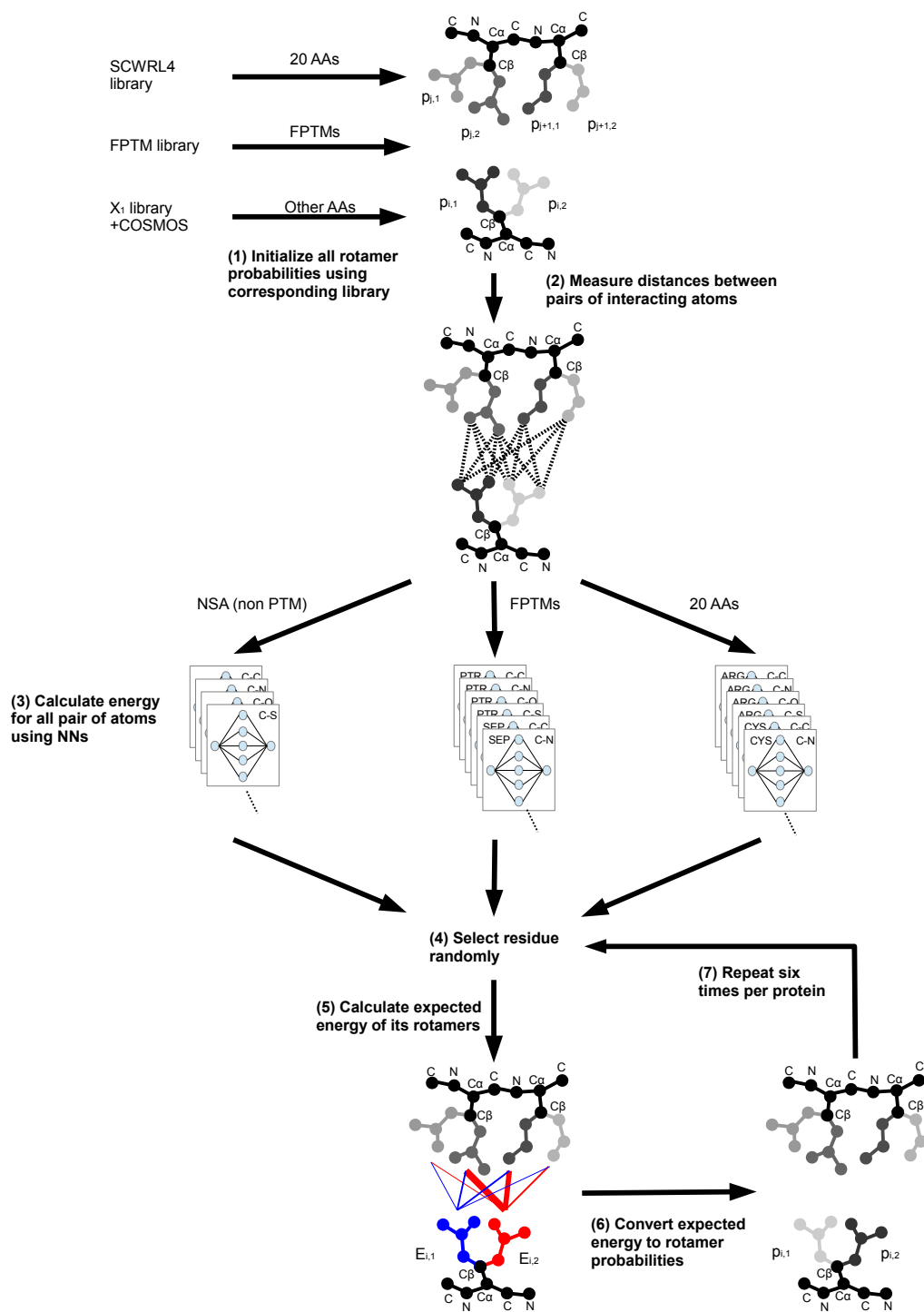


Figure 2.3: Prediction pipeline for optimizing all the rotamer probabilities. Once the optimization is completed, final predictions are produced by first selecting the most likely rotamer and then going through a clash reduction algorithm.

(120°,240°) and (240°,360°). The mean and standard deviation of the  $\chi_1$  angles for each rotamer bin were calculated. By default, the values of the  $\chi_i, i \geq 2$  angles are fixed to those of the original NSA structure. If a user provides multiple structures for a given NSA, then SIDEpro automatically builds a uniform rotamer library for  $\chi_i, i \geq 2$ . For the SIDEpro web server and downloadable program, we use the COSMOS program [3] for predicting the conformations of small molecules, to produce 10 conformations for each NSA that is not a FPTM.

### 2.3.3 Training Energy and Prediction

To predict side chains, SIDEpro uses an additive energy function parameterized using a large number of neural networks trained from the data. All the neural networks have identical structure with one input unit corresponding to a distance between a pair of atoms, one hidden layer with 15 sigmoidal hidden units, and one linear output unit computing the corresponding energy term as shown in Figure 2.1.

#### Neural Networks

For natural amino acids, there are 156 neural networks, one per amino acid type and per atom pair type. For the most frequent PTMs there are 130 new neural networks, one per FPTM type and per atom-pair type. Thus, for instance, there is one carbon-carbon neural network for phosphorylated Serine. For NSAs (non-FPTMs) we use a more generic approach with 25 neural networks, one per atom pair type. Note that as a slight simplification in all cases we consider only five atom types (C, H, N, O, S), treating P as if it were C, and using H only in the second position of an interaction.

## Training

The training pipeline is summarized in Figure 2.2. For a given protein in the training set with a fixed backbone, we initialize each rotamer to the value closest to the native conformation. Then we cycle once through each protein in the training set from the C-terminus to the N-terminus. When a given amino acid is being considered, we compute the energy of all its rotamers using the corresponding neural networks. These energies are converted into probabilities and then compared to the native conformation. The mismatch information is used to adjust the weights of the neural networks using Markov Chain Monte Carlo methods (see [61] for more details). For NSAs (non-FPTMs), we use the original SIDEpro training set (Table 2.2) of 252 proteins to train generic energy function neural networks using distances between pairs of atoms in all types of natural amino acids. The SIDEpro training set is a non-redundant dataset with SCWRL4 test dataset [45] with 25% sequence similarity. For FPTMs, the final downloadable and server version using neural networks trained on entire FPTM datasets. We set the hidden size which maximize the cross validated accuracy.

## Prediction

The prediction pipeline is summarized in Figure 2.3. In prediction, we are given a protein with a fixed backbone and possibly also a set of additional atoms with fixed coordinates, which typically correspond to fixed side chains or atoms in ligand molecules. For the remaining amino acids, we cycle through them in random uniform order without replacement. Each amino acid has its own library of rotamers and rotamer probabilities as described in Section 2.2. This is true for natural amino acids, for FPTMs, and for other NSAs initialized uniformly over 10 conformations produced by COSMOS. For a given non-fixed amino acid, we compute the expected energy of each one of its rotamers, given all the other fixed atoms, rotamers, and rotamer probabilities. These energy values are converted to probabilities and

the corresponding rotamer probability table is updated. The full cycle is repeated 6 times for each protein.

It is important to note that the neural networks are used only once to compute all the possible energy values, since the set of all possible pairwise distances, across all possible rotamer values, does not change during the prediction phase. For the final prediction, we choose the most likely rotamer configuration for each amino acid that is not fixed by the user. Finally, we run the same clash reduction algorithm as in previous section.

## 2.4 Results

### 2.4.1 Evaluation Data and Protocol

To conduct a comparative evaluation of SIDEpro and SCWRL4, the following datasets are used: (1) the benchmark dataset of 379 proteins used to evaluate SCWRL4 [45] (SCWRL4 dataset); (2) 94 proteins determined by X-ray crystallography and released in the most recent Critical Assessment of Protein Structure Prediction Experiment (CASP9 dataset); (3) a small set of seven large protein complexes (pdb: 1RYP, 2JES, 2UVB, 3GND, 3GZU, 3K1F, 3KQK) ranging in size from 2760 to 8767 residues (COMPLEXES dataset); and (4) a ribosome (pdb: 1FJG) with and without the RNA (RIBOSOME dataset).

In SCWRL4 the default predictor utilizes a Flexible Rotamer Model (FRM), but a Rigid Rotamer Model (RRM) is also available. The basic tradeoff between the two is that FRM is more accurate but slower than RRM [45]. Our preliminary tests of SCWRL4 confirmed this tradeoff, thus, both FRM and RRM are evaluated on each test dataset in this work and the summary results of both are compared to SIDEpro; however, only the FRM results are reported in the residue specific accuracy results for the SCWRL4 dataset (Table 2.17) and



the CASP9 dataset (Table 2.12).

The summary results tables present the  $\chi_1$  accuracy,  $\chi_{1+2}$  accuracy, and RMSD averaged over all residues, the average CPU time per protein, and the total number of severe clashes and moderate clashes. The summary results are presented as follows: SCWRL4 dataset in Table 2.9, CASP9 dataset in Table 2.11, COMPLEXES dataset in Table 2.13, and the RIBOSOME dataset in Table 2.14. In all tables the best result according to each metric is shown in bold.

## 2.4.2 Accuracy

Accuracy is assessed using three standard metrics: (1) percentage of side-chains where  $\chi_1$  is within  $40^\circ$  degrees of the experimental structure angles; (2) percentage of side-chains with both  $\chi_1$  and  $\chi_2$  within  $40^\circ$ ; and (3) root mean square deviation (RMSD), which is calculated using the absolute coordinates of the corresponding model and experimental structure side-chain atoms. For  $\chi_1$  evaluation, the symmetries of Phe and Tyr are accounted for by calculating  $\chi_1$  using both symmetric atoms and checking each result versus the experimental structure. For  $\chi_{1+2}$  evaluation the symmetries of Asp, Phe, and Tyr are accounted for similarly. In evaluating the RMSD the symmetries of Arg, Asp, Glu, Phe, and Tyr are accounted for by calculating the RMSD using both possible atom mappings and keeping the minimum result.

The accuracy summaries calculated on the SCWRL4 dataset are presented in Table 2.9. Overall, SIDEpro is slightly more accurate than SCWRL4-FRM according to all three accuracy measures:  $\chi_1$  (86.14% vs 85.43%),  $\chi_{1+2}$  (74.15% vs 73.47%), and RMSD (0.911 Å vs 0.948 Å). SCWRL4-RRM is the least accurate according to all three measures. Table 2.17 provides the residue specific accuracy results of SIDEpro and SCWRL4-FRM on the SCWRL4 dataset [45].

The accuracy summaries calculated on the CASP9 dataset are presented in Table 2.11. Again, SIDEpro is slightly more accurate than SCWRL4-FRM according to all three accuracy measures:  $\chi_1$  (85.01% vs 84.21%),  $\chi_{1+2}$  (72.76% vs 72.21%), and RMSD (0.940 Å vs 0.977 Å). Again, SCWRL4-RRM is least accurate according to all three measures. Table 2.12 provides the residue specific accuracy results of SIDEpro and SCWRL4-FRM on for the CASP9 dataset.

The accuracy summaries calculated on the COMPLEXES dataset are presented in Table 2.13. On this set SIDEpro is clearly more accurate than SCWRL4-FRM according to all three accuracy measures:  $\chi_1$  (79.73% vs 77.99%),  $\chi_{1+2}$  (64.19% vs 62.57%), and RMSD (1.109 Å vs 1.184 Å). Again, SCWRL4-RRM is least accurate according to all three measures.

The accuracy summaries calculated on the RIBOSOME dataset with and without RNA are presented in Table 2.14. When the predictions are made without the RNA coordinates provided to the methods SIDEpro is more accurate than SCWRL4-FRM according to all three accuracy measures:  $\chi_1$  (71.50% vs 70.61%),  $\chi_{1+2}$  (53.60% vs 52.37%), and RMSD (1.504 Å vs 1.559 Å). SCWRL4-RRM is the least accurate according to all three measures. When the predictions are made in the presence of the RNA coordinates the accuracies of all three methods improve according to all three measures. SIDEpro is still the most accurate, followed by SCWRL4-FRM, with results of  $\chi_1$  (74.30% vs 72.31%),  $\chi_{1+2}$  (56.00% vs 54.49%), and RMSD (1.423 Å vs 1.480 Å). SCWRL4-RRM is still the least accurate according to the three measures.

### 2.4.3 CPU Time

Figure 2.4 shows the relationship between the number of residues in a protein and the prediction time for SIDEpro, SCWRL4-FRM, and SCWRL4-RRM using the 379 proteins from the SCWRL4 dataset. For SIDEpro, the time follows a predictable linear increase

according to the number of residues with a Pearson correlation of 0.97. For both SCWRL methods, the time generally increases with the number of residues, but the relationship is much less predictable. The Pearson correlation for SCWRL4-RRM is 0.53 and for SCWRL4-FRM it is 0.48. On the SCWRL4 dataset the average CPU time needed by each method to make prediction is 1.61s for SIDEpro, 11.09s for SCWRL4-FRM, and 3.84s for SCWRL4-RRM (Table 2.9). The gap in CPU time between SIDEpro and SCWRL is more significant on the CASP9 dataset where the average CPU times are 2.06s for SIDEpro, 19.06s for SCWRL4-FRM, and 10.64s for SCWRL4-RRM (Table 2.11). For both of these datasets the times are calculated using all protein chains in each PDB file as input. The CPU times for all experiments are obtained using an AMD Turion 64 X2 Mobile Technology TL-60+ at 2.00 GHz, with 3.00 GB of RAM, and running 32-bit Microsoft Windows Vista Home Premium Service Pack 2.

For the COMPLEXES dataset the gap between the CPU times required by the different methods is much more significant. The average CPU times are 26.9s for SIDEpro, 583.9s for SCWRL4-FRM, and 409.4s for SCWRL4-RRM (Table 2.13). For this dataset the times are calculated using the biological assemblies as input. On the RIBOSOME dataset without the RNA the CPU times are 11.8s for SIDEpro, 180.0s for SCWRL4-FRM, and 102.1s for SCWRL4-RRM. When the RNA coordinates are provided the CPU time for SIDEpro increases only slightly to 13.7s; in contrast the time jumps to 4020.7s for SCWRL4-FRM and 796.2s for SCWRL4-RRM (Table 2.14). Note that all three methods exhibit similar increases in accuracy when the RNA coordinates are utilized in prediction. In sum, once trained, SIDEpro is significantly faster than the other programs.

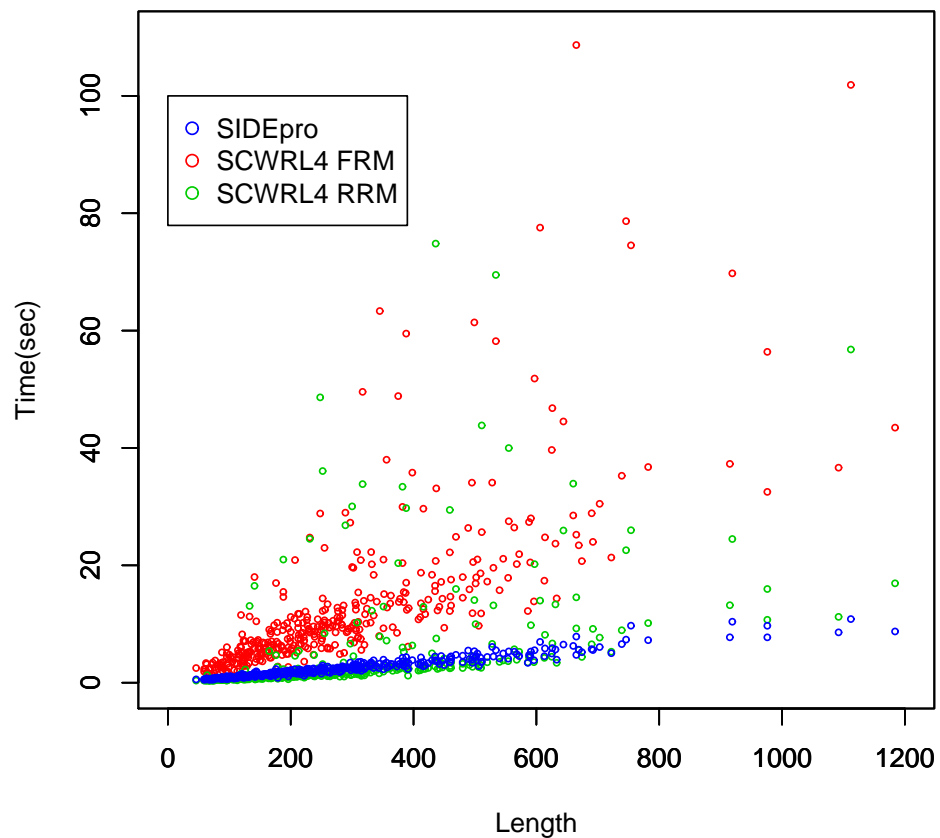


Figure 2.4: CPU Times versus Protein Length for the 379 Protein SCWRL4 Dataset. The CPU time required by SIDEpro increases linearly with the number of residues, with a Pearson correlation of 0.97. For both SCWRL methods, the CPU time generally increases with the number of residues, but the relationships are less predictable. The Pearson correlation for SCWRL4-FRM is 0.53 and for SCWRL4-RRM it is 0.48.

#### 2.4.4 Clash Assessment

Two types of clashes are defined here and used for assessment: severe and moderate. In order to assess the frequency of clashes appropriately and to make fair comparisons we utilized SCWRL parameters related to steric repulsion to define clashes.

The SCWRL repulsive energy term applies no penalty if the observed distance,  $d$ , between atoms  $i$  and  $j$  is greater than the sum of their hard sphere radii ( $vdw_{ij}$ ). The maximum penalty is applied if  $d/vdw_{ij} < .8235$ , and a linear ramp is used if  $.8235 \leq d/vdw_{ij} \leq 1$  [13, 15, 45]. Thus, we define a moderate clash to occur if  $.8235 \leq d/vdw_{ij} \leq 1$ , and a severe clash if  $d/vdw_{ij} < .8235$ . Clashes are counted at the level of residue pairs and only the minimum observed  $d/vdw_{ij}$  for each residue pair is considered. This means that each residue pair can only be counted as one of: (1) unclashed, (2) moderate clash, or (3) severe clash. The radii used in this assessment are the same as those used in the SCWRL steric energy: carbon, 1.6 Å; oxygen, 1.3 Å; nitrogen, 1.3 Å; and sulfur 1.7 Å.

The counts of severe and moderate clashes observed in the models produced by SIDEpro, SCWRL4-FRM, and SCWRL4-RRM are presented in the summary tables for each dataset. The clashes observed in the corresponding PDB structures are also included for comparison. In general, the disparity in the number of clashes between the PDB structures and the models produced by all methods demonstrates that there is still room for improvement in terms of clash resolution.

On the SCWRL4 dataset the SIDEpro models have fewer severe clashes (305 vs 1047) and fewer moderate clashes (7227 vs 9743) than SCWRL4-FRM models, and also fewer severe clashes than SCWRL4-RRM models (496); however, SCWRL4-RRM has the smallest number of moderate clashes (6661).

On the CASP9 dataset the pattern is repeated. SIDEpro models have fewer severe clashes

(112 vs 356) and fewer moderate clashes (2403 vs 3170) than SCWRL4-FRM models, and also fewer severe clashes than SCWRL4-RRM models (172); however, SCWRL4-RRM has the smallest number of moderate clashes (2117).

On the COMPLEXES dataset the pattern is repeated again. SIDEpro models have fewer severe clashes (253 vs 972) and fewer moderate clashes (6114 vs 7493) than SCWRL4-FRM models, and also fewer severe clashes than SCWRL4-RRM models (540); however, SCWRL4-RRM has the smallest number of moderate clashes (5490). This same pattern is also repeated on the RIBOSOME dataset both with and without RNA.

### 2.4.5 Generic Energy vs Amino Acid Specific Energy

The generic neural networks and the corresponding energy can first be tested on the 20 natural amino acids and compared to the amino acid specific neural networks of SIDEpro. Comparison of these two approaches on the SCWRL4 test set, using the SCWRL4 rotamers[45], are reported in Table 2.15, with a summary for each amino acid of the RMSD, the average  $\chi_1$  and average  $\chi_{1+2}$ , and the corresponding  $p$  values for a paired  $t$  test on the RMSD. For each metric and each amino acid, the best results are shown in bold together with all  $p$  values that are less than 0.15. When all amino acid types are considered as a single large test set, the amino acid specific neural networks produce slightly more accurate models according to all three metrics with high significance ( $p < 0.001$ ). For ten amino acid types, the amino acid specific neural networks perform better than the generic neural networks significantly  $p < 0.03$ . Note that the generic neural networks produce better results for all three metrics for Tyrosine and Phenylalanine, and for at least one of the three metrics for four other residue types. However, these differences are not statistically significant since there is no amino acid type for which the generic neural networks perform better at a significant level  $p < 0.15$ . Taken together these results show overall that: (1) as expected, the amino

acid specific neural networks perform better than the generic neural networks on the natural amino acids; (2) the generic neural networks are not far behind, with RMSDs below  $1\text{\AA}$  most of the time, and provide reasonable models and a reasonable alternative, with considerably less parameters.

### 2.4.6 Prediction of FPTMs

Here we compare the performance of the FPTM specific neural networks and the generic neural networks for the prediction of FPTMs. One FPTM type, PCA, is excluded from the comparison since it has only one rotamer. We used five-folds cross validation on FPTM datasets. Table 2.16 shows the average number of atom pairs used for training the FPTM specific neural networks, the number of neural networks, the ratio of these two numbers, and the corresponding cross validated RMSDs and  $p$  values for a paired  $t$  test on RMSDs of each fold. The best RMSD values are in bold together with  $p$  values less than 0.15. The number of training atom pairs divided by the number of neural nets provides a rough estimate of the number of examples used for training the neural networks of each FPTM. For four FPTM types (LLP, MLY, PTR and MSE), the FPTM specific neural networks perform better than the generic neural networks with significance  $p < 0.15$ . These four types are one of the five highest values of the average number of training pairs per neural network. For all of PTMs except OCS and SEP, the specific neural networks perform better, but the difference is not significant. For two PTM types (OCS and SEP), the generic neural networks perform better with no significance. Since none of PTM shows better performance with statistical significance, we decided to use the specific energy for all FPTMs.

Table 2.17 summarizes the cross validated prediction accuracy results for the FPTMs, grouped according to their precursor amino acid, on the FPTM datasets. Each precursor amino acid is shown in bold together with the corresponding SIDEpro results. The table

presents the average number of instances observed in the test set for each of the 15 FPTMs, as well as the cross validated results of the three accuracy metrics (RMSD,  $\chi_1$ , and  $\chi_{1+2}$ ). Two average RMSD results are presented using: (1) only the atoms in common with the precursor amino acid; and (2) all the atoms. The former allows for a direct comparison with the accuracy of SIDEpro on the precursor amino acid.

Considering the RMSD metric, and only atoms shared with the precursor, the accuracy for the FPTM is somewhat comparable to the accuracy of SIDEpro on the precursor amino acid. In fact, four PTMs have lower mean RMSD than their precursor: KCX-Lysine, M3L-Lysine, MLY-Lysine and MSE-Methionine. When all the atoms in the PTM amino acid are considered, the average RMSD results are significantly higher. This results directly from the increase in size and degrees of freedom of each PTM amino acid with respect to its precursor amino acid. Considering the  $\chi_1$  metric, six of the FPTMs have higher accuracy values than their precursor amino acid: CSD-Cysteine, KCX-Lysine, LLP-Lysine, M3L-Lysine, MSE-Methionine, and PCA-Proline. Six other PTMs have  $\chi_1$  accuracy that is within 10% of the corresponding precursor amino acid result. Considering the  $\chi_{1+2}$  metric, only the PTMs associated with Lysine, Tyrosine, Methionine, and Proline can be compared. Out of the eight corresponding PTMs where a direct comparison to the precursor atoms can be made, four have higher accuracy values than their precursor: KCX-Lysine, M3L-Lysine, MSE-Methionine, PCA-Proline. In short, by multiple metrics, the prediction accuracy of SIDEpro on the 15 FPTMs is roughly comparable to its accuracy on the natural amino acids, including by the RMSD metric when only atoms in common with the precursor are considered. A significant but expected deterioration in RMSD metric is observed when all atoms are considered, due to the increase in the size and number of rotatable bonds associated with PTMs.



### 2.4.7 Prediction of NSAs

The generic NSA prediction method requires a 3D structure model of the NSA be provided as input and in order to test the NSA method with more data we tested it on both the FPTM and the NSA (non-FPTM) test sets. Structure models are derived from two sources: (1) true structures from the PDB; and (2) conformations generated by COSMOS [[3]]. Results obtained using true structures do not reflect what can be expected from prediction in a realistic setting, but rather provide a sense of the limits of the methods. In true prediction mode, the structure of the NSAs must be generated by a small molecule structure predictor.

Table 2.18 presents the results of the generic NSA prediction method on the FPTM set, when the FPTM amino acids are treated as non-standard. The best results for each metric and each FPTM are in bold. In this experiment, for each modified amino acid, we use a single predicted structure obtained with COSMOS. As shown in the next section, further improvements can be obtained by using multiple predicted structures. As expected, with a few exceptions, when the true structures are used as input the resulting models are more accurate than when predicted structures are used as input. Overall, the predicted structures lead to reasonable performance, given the complexity of the problem and the high-throughput nature of the approach. In all cases using predicted structure leads to RMSD values that are always below  $2.5\text{\AA}$  on the atoms shared with the precursor amino acids.

Finally, Table 2.20 summarizes the results obtained on the NSA (non-FPTM) test set. For this experiment, we compare the results obtained using the true structure from the PDB, a single predicted structure, and multiple (10) predicted structures as structural models for the NSAs. As expected, using the true structure provides the most accurate results, with an average RMSD of  $1.75\text{\AA}$  and a  $\chi_1$  of 66.63%. Using multiple predicted structures helps improve the performance. For instance, the average RMSD improves from 3.54 to  $3.08\text{\AA}$ , a value that is reasonable given the high-throughput nature of the approach and the complexity

and variability of NSAs, but requiring further refinements for high-precision tasks. In terms of the  $\chi_1$  metric, using 10 structures improves the performance from 56.39% to 65.30%, a value very close to the performance obtained using the PDB structures.

## 2.5 Discussion

Methods that can predict protein side-chains quickly and accurately can be applied in many important applications in computational molecular biology. In this investigation, we have described a new machine learning approach to the problem, resulting in a new predictor SIDEpro, and compared its performance to SCWRL4 on various datasets. In all cases, we found the same basic result: SIDEpro is more accurate and faster than both versions of SCWRL4. Table 2.21 summarizes the statistical significance of SIDEpro's improvements in accuracy, with respect to SCWRL4-FRM and SCWRL4-RRM, for each dataset using each metric. Considering the results on the large datasets, SCWRL4, CASP9, and COMPLEXES, the difference between SIDEpro and SCWRL4-FRM accuracies is significant at  $p < 0.001$  for all results except CASP9  $\chi_{1+2}$  where  $p = 0.05$ . When comparing SIDEpro and SCWRL4-RRM on the large datasets, all of the accuracy results presented are significant at  $p < 1.0e-10$ .

When compared to the SCWRL4 Free Rotamer Model SIDEpro represents a moderate, but statistically significant improvement in accuracy, but the difference in CPU time is significant: SIDEpro is from 7 to 20 times faster depending on the dataset. When compared to the SCWRL4 Rigid Rotamer Model the improvement in accuracy is more significant, and SIDEpro is still from 2 to 15 times faster.

We have extended the capabilities of SIDEpro to PTMs and NSAs. For natural amino acids and very frequent PTMs, SIDEpro uses amino acid specific energy functions. In order to flexibly accommodate for any non-standard amino acid, SIDEpro allow users to provide 3D

structures of NSAs to be incorporated into SIDEpro models. Alternatively, the COSMOS [3]) program is used to predict these structures, and any other similar program (e.g. OpenBabel [64]) can be used for the same purposes. The generic neural networks, trained on all possible pairs of atom types agnostic of residue type, are used to score the atom-atom interactions for these NSAs. Naturally, as more data on NSAs becomes available in the PDB, it will be possible to further expand the set of specific energy functions, thereby increasing the accuracy of the program over time. As demonstrated here for some of the NSAs, accuracy can also be improved by increasing the number of 3D samples produced by COSMOS, at the expense of time.

Finally, SIDEpro is to be used in protein structure prediction and engineering projects for the rapid prediction of side chains conformations in high-throughput mode, or to provide good starting points for molecular or quantum mechanics simulations of side chain atoms, for both standard and non-standard amino acids.

Table 2.5: FPTM dataset

PDB Ligand ID	PDB IDs
ABA	1A3P, 1B6J, 1BAH, 1BBO, 1C2U, 1CSA, 1CYA, 1CYB, 1IKF, 1K09, 1RBD, 1T1Q, 1T9E, 1WZ5, 1ZII, 2ESL, 2M1P, 2M2G, 2M2H, 2M2S, 2M2X, 2X7K, 2ZOK, 3FSM, 3LO6
CME	1A1V, 1EUD, 1HQS, 1JN9, 1PME, 1PZG, 1Q9U, 1QQQ, 1YKM, 2AQ5, 2CZL, 2J83, 2P3A, 2Q59, 2QXS, 2V0Y, 2Y3C, 2Y6V, 3CFA, 3D6I, 3GUI, 3IXL, 3KK4, 3N5G, 3NHE, 3T5W, 3UWL, 4A1K, 4B8X, 4BGP, 4D9G, 4GOA, 4MPB
CSD	1DIN, 1DNC, 1EU1, 1FNK, 1OCH, 1Q1Y, 1Q78, 1QMV, 1RT1, 1SUB, 1UKK, 1VGX, 1VI9, 1XW4, 2BC0, 2DD5, 2E2M, 2EG3, 2HAI, 2OPL, 2P0U, 2PVZ, 2QUG, 2R3I, 2RIL, 3A9C, 3BC2, 3EUO, 3F5V, 3H56, 3OQI, 3P5U, 3PSZ, 3SED, 3SQZ, 3TKS, 3VYH, 3WEU, 4EKF, 4FLM
CSO	1DMP, 1EQ2, 1G55, 1GSN, 1I9T, 1JOA, 1JZ7, 1K3I, 1LG7, 1LOQ, 1O4C, 1O8V, 1O9Q, 1OET, 1PL1, 1PRX, 1Q79, 1RQ4, 1SVY, 1W2M, 1W6M, 1WL4, 1WL8, 1XCM, 1XVW, 1Y1F, 1YJA, 1YML, 2AOU, 2AQ5, 2BFZ, 2BJA, 2CIR, 2CVO, 2D1Q, 2DD5, 2EG4, 2F2L, 2FHJ, 2FHX, 2HCJ, 2HP0, 2HPR, 2ID4, 2ISY, 2J89, 2NQA, 2ORA, 2PVJ, 2QTZ, 2QX0, 2RFV, 2RG2, 2VH3, 2VR8, 2VRN, 2WAW, 2X2H, 2XF1, 2ZCT, 3AAY, 3B4Y, 3B8B, 3BB0, 3BOO, 3BQG, 3C6B, 3CIW, 3CKC, 3CV2, 3D3W, 3DQY, 3F71, 3FE5, 3FSG, 3HRM, 3IB4, 3IV0, 3KEV, 3KMZ, 3LAC, 3MII, 3NA8, 3NON, 3OT4, 3P5V, 3QD5, 3QSB, 3QSR, 3RJT, 3SUJ, 3TU8, 3U11, 3U1P, 3U2A, 3U7E, 3UBW, 3UUC, 3VYH, 3ZVH, 4ASC, 4AZ4, 4BG4, 4BGC, 4C5P, 4DFE, 4DWN, 4EO7, 4EOC, 4EVX, 4FST, 4GE7, 4GQZ, 4HTF, 4JIH, 4LIX, 4PGT
HYP	1AG7, 1AS5, 1DLZ, 1EYO, 1G1P, 1GQ0, 1IEO, 1IH9, 1JLP, 1JOH, 1K64, 1KCP, 1MTQ, 1OB4, 1OB6, 1OB7, 1P1P, 1PQR, 1Q7D, 1R9I, 1R9U, 1TCG, 1UW9, 1VIB, 1WCT, 2B5P, 2B5Q, 2CCO, 2EW4, 2H9X, 2IH6, 2IH7, 2IHA, 2J15, 2JQB, 2JQC, 2JRY, 2JTU, 2KLW, 2LAQ, 2M32, 2VUV, 2W65, 2YYF, 3P46, 3POD, 3U29, 4G13, 4G14

Table 2.6: FPTM dataset

PDB Ligand ID	PDB IDs
KCX	1BWV, 1E8C, 1JBV, 1L6F, 1M6K, 1OIR, 1ONW, 1PU6, 1RQB, 1W78, 2FVM, 2GWN, 2ICS, 2JFF, 2P9V, 2QF7, 2QPX, 2UYN, 2WTZ, 3C0Q, 3HBR, 3JZE, 3KZN, 3LA4, 3MTW, 3NWR, 3OVG, 3PNZ, 3TN3, 4C12, 4GN2, 4MWA
LLP	1AX4, 1BJN, 1BW0, 1D7K, 1IUG, 1S06, 1YIZ, 2BWN, 2CB1, 2FM1, 2GB3, 2OKJ, 2PYD, 2QLR, 2VYC, 2W8T, 2X3L, 3B46, 3BC8, 3BWO, 3CEB, 3CQ6, 3F0H, 3F6T, 3G0T, 3GWP, 3H7F, 3HA1, 3IAU, 3JTX, 3JU7, 3K40, 3KE3, 3KW3, 3L44, 3L6R, 3LLX, 3LUL, 3NNK, 3NRA, 3NYU, 3OP7, 3PIU, 3PJ0, 3QQM, 3R4T, 3SPX, 3U9X, 3VAB, 4DZA, 4E3R, 4F4E, 4H27, 4IXO, 4JE5, 4LHC
M3L	1CCR, 1PDQ, 1PRW, 2B2U, 2F6J, 2G6Q, 2GFA, 2H6Q, 2JMJ, 2K17, 2L11, 2L3R, 2L75, 2LBM, 2LGK, 2M00, 2OQ6, 2RR4, 2RSN, 2V83, 2X4W, 2YK3, 3AVR, 3C6W, 3GL6, 3GV6, 3JPX, 3KQI, 3KV4, 3LQJ, 3M5A, 3ME9, 3MP1, 3O7A, 3QL9, 4BD3, 4HON, 4L58, 4L7X
MLY	1GUW, 1KNA, 1LLN, 1R1G, 1VK1, 2F4I, 2FSA, 2H13, 2KVM, 2LVM, 2QHQ, 2RFI, 2RHI, 2V88, 2VD9, 2VPE, 2ZPM, 3BED, 3C0F, 3F9X, 3KV4, 3LM9, 3LN3, 3LTI, 3M56, 3MC3, 3MET, 3MP6, 3NIO, 3QDP, 3QWZ, 3TZD, 3V0S, 3VWW, 4AU7, 4DWS, 4EE6, 4GNR, 4IQ0, 4JDU, 4KM8
MSE	4GHN, 3I7M, 2DC0, 3QVQ, 3CQ1, 2J43, 3FLK, 3VCX, 2V6V, 1K8U, 2Z07, 3TOS, 2QSX, 3NRE, 2GUH, 1RXD, 2ORD, 1J58, 3S83, 2RGQ, 2RA9, 2BHY, 3KGY, 3CGH, 2EWR, 3K0B, 3D33, 1H65, 2QNT, 2NQW, 3QOM, 3OF5, 4EQ8, 3L1W, 1ZMA, 1UWW, 3BWL, 2VRZ, 3OUV, 2QFF, 1Z67, 3EBT, 2ICP, 1U61, 2NYI, 4FR9, 3BJD, 3KBY, 2NL9, 2EVE, 3K1Z, 1AZO, 2I9W, 2QHQ, 1J2V, 3KA7, 3OIO, 1QX0, 3LQ9, 3H36, 1FBN, 1UI5, 2YZ8, 3S3T, 3O46, 4HUJ, 2O57, 4I8I, 2INW, 4LII, 1U7H, 4EBG, 3F67, 3CHV, 2QLX, 1RRE, 4KT3, 3GVZ, 2GHS, 2OQK, 3QUF, 3EC9, 3MCW, 4M8K, 2QV8, 3G5T, 2EVR, 3VPI, 1K7K, 3I0Y, 3H0N, 3NNB, 3H75, 3FSD, 3CED, 1JB6, 3EJK, 3N0X, 1P3D, 3C8M, 4EAE, 3KWR, 3FCD, 3JQ1, 1X99, 1Ouo, 1IXC, 2X9X, 4LBA, 3AL2, 3K7C, 2ZPL, 3GO9, 2OZH, 3U54, 2CVB, 3I0Z, 4MU9, 3K0T, 3FUT, 3DMC, 3CIT, 3BCZ, 3MAJ, 1MFW, 1YVO, 1UPG, 1MPX, 3CCG, 1LR0, 4LZK, 3RPD, 2UVK, 3UF6, 3GMY, 1WK2, 2YH9, 2C2I, 3IB5, 3F40, 3BLZ, 1UUH, 3KWK, 2RHF, 1KGS, 3MQO, 3L22, 4IAU, 3E0X, 3KKG, 1US4, 2HTD, 3E9V, 3P02, 3H4Y, 1DNL, 3KHN, 3CP0

Table 2.7: FPTM dataset

PDB Ligand ID	PDB IDs
MSE	3NPF, 3IHV, 3MFN, 3EXQ, 1J4J, 3R13, 3DCZ, 3BKX, 2RBB, 3FJ2, 2Q0Q, 4ECF, 4IAG, 2W1S, 2RAU, 2R0X, 1O3U, 3EPV, 2FB0, 3OXP, 3III, 1YOD, 2FPD, 3M6Y, 3A10, 2III, 3GJU, 3OFG, 3IEH, 1XY7, 3PFE, 3BM3, 4N0R, 2PN0, 3O0W, 2ODA, 1VAJ, 1PBJ, 3C0S, 2GKP, 4ACY, 2PRV, 1UKK, 3HTL, 3KOG, 3H71, 3BU9, 2R2Z, 1O1X, 3FG8, 3ZQO, 1DFM, 2W3D, 2O0P, 1FG3, 1Y9B, 2OT9, 1R43, 3C9Q, 1SC0, 2NLV, 3OWR, 3LJI, 4BWR, 4ID0, 3KAO, 2EXR, 1DOW, 4EW7, 3SEE, 4HKF, 2IA4, 3NBM, 3ODT, 3K12, 1ZGK, 1FVG, 2CWZ, 3CK2, 2OEE, 3D1P, 1DUS, 3MSR, 1VHU, 4EW5, 1KKO, 3DMW, 3I10, 2I8D, 3L51, 3DCY, 4HDE, 3MCX, 3S6F, 1CT5, 2X9Z, 1UIX, 3DZ1, 3E23, 2FHQ, 2XIW, 3LWJ, 2AKO, 2FG1, 3I4G, 2WML, 3OOU, 2P3P, 2D4O, 2A5L, 2R8W, 4F0J, 2O55, 2W7V, 3PAN, 2OP5, 2O2X, 3KIZ, 2IOC, 3KE7, 2IAI, 2VQC, 1Y81, 3FZ4, 3IUO, 3OOS, 1TZA, 1JI7, 2R6V, 1XHD, 2D9R, 1KR4, 3B5E, 2ODI, 3TD9, 2RK9, 3LYD, 2WVX, 3LM3, 1WGB, 1A7A, 2HO4, 2QHP, 3EC6, 3OLQ, 2GQ1, 1FIM, 3MCQ, 1YQ5, 2GS5, 1NS5, 3IKB, 3ZIE, 3R6D, 2A5Z, 3BGU, 4FXV, 1K7J, 4A3Z, 3UWB, 2OGF, 2HLZ, 3H9M, 2UV0, 4I95, 3I09, 2EH3, 1ZS9, 3HVV, 1IXL, 3TLG, 3GXX, 3I2V, 1QBQ, 1ZKO, 4IYH, 4JWO, 3K11, 3KNW, 3OOX, 3KA5, 2BZ1, 3FWZ, 2QNG, 3LD7, 3CJY, 3H74, 3D5P, 3MWZ, 2QMA, 3SGG, 2BSH, 3G14, 2ATF, 4FS7, 3GYK, 3M7A, 3MC3, 2QEU, 3KC2, 4KQC, 3PJY, 3S9X, 2FYK, 2QRU, 1YI7, 2Q2X, 2R0S, 3SN0, 3K9I, 4GHJ, 3ZXJ, 2WOY, 2I2O, 3OSD, 2OYZ, 4MLZ, 4EZI, 3FG9, 4HCF, 2QXY, 3ECF, 1M1S, 3HYN, 3CI6, 3K0Z, 1T5H, 1L8R, 1Y5H, 3D4E, 3RJV, 4M1Q, 1UWN, 2XQO, 3KZT, 1ZK8, 3BWS, 2A13, 3KYZ, 1Z7A, 2Q4X, 3HN0, 3JYB, 2RLC, 3CBT, 3DD7, 2IHT, 3PT1, 2WQ4, 3EY5, 3HEB, 2NXF, 3JYG, 3DUL, 1PMH, 3KPA, 5FIT, 2QSI, 3IWF, 4E6F, 2NMU, 3E6Q, 3BCY, 2RIL, 4J4Z, 3G5S, 4K05, 2I5U, 2YG8, 3LOP, 2HXI, 2CXY, 1VHF, 3LWC, 3UE2, 3DV9, 1MGP, 3IEE, 3H96, 3EXN, 4G68, 1QWX, 3E8O, 3EN8, 3QXF, 2PEB, 3G23, 2V9D, 3EJV, 3K1U, 3ER7, 2IMH, 2P2R, 3E4V, 3LCU, 4GJY, 2FA8, 1KLL, 3AZO, 2CDO, 4JVT, 3POH, 3EH7, 3FW2, 3GR3, 2QU1, 2OOC, 3G36, 1Y0B, 1Y0K, 3O6C, 2QSB, 1QO2, 1VJU, 3HRP, 4JJA, 3OHE, 1TR9, 2O0M, 4H4J, 2WLR, 1YLE, 1Z42, 2Z0U, 4G5A, 3CGG, 3B79, 3GBY, 2DXQ, 3S8M, 2P42, 2QNK, 4IRH, 4GD5, 4HMP, 2IAY, 3U97, 3GHJ, 3MZ2, 4K4K, 3LLX, 3SY6

Table 2.8: FPTM dataset

PDB Ligand ID	PDB IDs
OCS	1CS8, 1E6Y, 1HAV, 1J98, 1LIC, 1LME, 1MZS, 1NHS, 1O1X, 1O1Y, 1OEO, 1V3Y, 1Y1G, 1YMD, 1ZB8, 2CIO, 2CYJ, 2F1K, 2GPC, 2HHF, 2HL9, 2ILU, 2P8E, 2PN0, 2PT0, 2R47, 2RLC, 2UYJ, 2V1M, 2WFI, 2Y8B, 2ZZE, 3C4B, 3EF2, 3EIT, 3FBX, 3FGR, 3GS9, 3IU6, 3KOM, 3MC4, 3O5A, 3OMD, 3P9S, 3Q3X, 3QQD, 3RH0, 3ZVH, 3ZXO, 4AIW, 4FR7
PCA	1AYJ, 1BAH, 1BOM, 1BRZ, 1BUS, 1C4E, 1CGN, 1COR, 1CP9, 1D7C, 1DTX, 1EHD, 1FJ0, 1GQ8, 1H4H, 1KB3, 1KFP, 1KM8, 1KUG, 1MXQ, 1OE1, 1OFL, 1OLR, 1P9G, 1Q2J, 1Q8O, 1QI9, 1QOZ, 1R9I, 1RCK, 1S8K, 1THG, 1WGT, 1YM0, 1YY1, 2AXK, 2BNJ, 2CIV, 2GFR, 2K1V, 2KBC, 2LJS, 2LQA, 2LYW, 2OYV, 2OYW, 2PSP, 2XSP, 2YEN, 2Z49, 3C9X, 3E80, 3G2Y, 3KQ0, 3O8Q, 3T0V, 3VLA, 3ZYP, 4APJ, 4BUH, 4GFT, 4JP6, 5ACN
PTR	1AD5, 1AYB, 1BMB, 1CSY, 1D4W, 1EEN, 1FHR, 1H9O, 1IRS, 1J4X, 1JU5, 1K4T, 1KA6, 1LCJ, 1M0V, 1PKG, 1QG1, 1SHC, 1TCE, 1YRK, 2BBU, 2CI9, 2DVJ, 2H7D, 2HDX, 2I6O, 2IUH, 2L4K, 2LCT, 2LNW, 2LQW, 2NMB, 2OQ1, 2Q8Y, 2QO7, 2RMX, 2ROR, 2VIF, 3BUM, 3EB0, 3GB2, 3KVV, 3MAZ, 3OLR, 3SAY, 3U3Z, 3VRO, 4DWP, 4GFU, 4GVC, 4K45
SEP	1B4G, 1GZ2, 1H4X, 1HJK, 1LWN, 1MKI, 1P5D, 1T6R, 1VKL, 1VRV, 2AFF, 2AZM, 2CEF, 2CEZ, 2CFJ, 2FEP, 2FWN, 2G57, 2JPW, 2KMD, 2LOI, 2L5J, 2LAJ, 2LAX, 2LB0, 2LIC, 2LID, 2LO6, 2LXT, 2M3B, 2PUM, 2Q0N, 2W3O, 3D9N, 3F3Z, 3GA7, 3HRC, 3L41, 3MK0, 3OB2, 3P35, 3Q4A, 3QPD, 3SHV, 3TMP, 3TPV, 3UBW, 3ZI7, 4BJU, 4FIH, 4HJH, 4IAC, 4ICD, 4IGK, 4N6Y, 4PEP
TPO	1G6G, 1GXC, 1I8G, 1J4L, 1J4P, 1J4X, 1V50, 2AFF, 2ERK, 2FF4, 2JOC, 2JQL, 2K7L, 2KFU, 2KMD, 2LAX, 2LB2, 2PIE, 2Q5A, 2Q8Y, 2RLT, 2W3O, 2W8D, 3AL3, 3BZI, 3E6Y, 3MVJ, 3OB1, 3OUN, 3POA, 3Q52, 3UNN, 3VA4, 4BU0, 4JG1, 4KAV, 4LR7

Table 2.9: SCWRL4 Dataset Summary Results: 379 Proteins, 58229 Residues.

Method	$\chi_1$ (%)	$\chi_{1+2}$ (%)	RMSD (Å)	Time(s)	Severe Clashes	Moderate Clashes
PDB					31	2048
SIDEpro	<b>86.14</b>	<b>74.15</b>	<b>0.911</b>	<b>1.61</b>	<b>305</b>	7227
SCWRL4-FRM	85.43	73.47	0.948	11.09	1047	9743
SCWRL4-RRM	84.15	71.24	0.995	3.84	496	<b>6661</b>

Table 2.10: SCWRL4 Dataset Residue Specific Accuracy Results.

AA Type	AA Count	$\chi_1$ (%)		$\chi_{1+2}$ (%)		RMSD (Å)	
		SIDEpro	SCWRL4	SIDEpro	SCWRL4	SIDEpro	SCWRL4
ARG	3636	<b>78.8</b>	78.3	65.2	<b>66.0</b>	<b>2.24</b>	2.33
ASN	2883	<b>85.4</b>	83.9	<b>61.6</b>	59.5	<b>1.04</b>	1.09
ASP	4018	<b>84.8</b>	84.0	75.3	<b>75.4</b>	<b>0.81</b>	0.82
CYS	1001	<b>90.4</b>	90.3			0.48	<b>0.47</b>
GLN	2512	<b>79.1</b>	78.7	<b>60.9</b>	58.9	<b>1.64</b>	1.69
GLU	4644	<b>73.2</b>	72.8	<b>57.6</b>	56.2	<b>1.47</b>	1.50
HIS	1543	<b>89.6</b>	87.9	<b>53.4</b>	50.7	<b>1.27</b>	1.35
ILE	3968	95.6	<b>96.0</b>	83.4	<b>84.7</b>	0.45	<b>0.43</b>
LEU	6558	<b>93.5</b>	92.1	<b>86.5</b>	86.2	<b>0.56</b>	0.58
LYS	3901	<b>79.3</b>	76.9	<b>66.5</b>	63.3	<b>1.63</b>	1.73
MET	1410	<b>83.8</b>	83.8	<b>74.5</b>	70.3	<b>1.14</b>	1.27
PHE	2717	<b>95.7</b>	95.3	<b>92.7</b>	92.6	<b>0.65</b>	0.72
PRO	3233	83.7	<b>85.5</b>	80.2	<b>81.6</b>	0.27	<b>0.26</b>
SER	4107	<b>74.0</b>	70.8			<b>0.73</b>	0.80
THR	3790	<b>90.8</b>	90.5			<b>0.39</b>	0.39
TRP	979	91.6	<b>92.1</b>	<b>82.6</b>	79.5	<b>1.16</b>	1.37
TYR	2346	94.5	<b>94.7</b>	91.0	<b>92.1</b>	<b>0.83</b>	0.86
VAL	5019	<b>93.2</b>	93.1			0.32	<b>0.32</b>
<b>All</b>	<b>58265</b>	<b>86.14</b>	85.43	<b>74.15</b>	73.47	<b>0.911</b>	0.948

Table 2.11: CASP9 Dataset Summary Results: 94 Proteins, 17885 Residues.

Method	$\chi_1$ (%)	$\chi_{1+2}$ (%)	RMSD (Å)	Time(s)	Severe Clashes	Moderate Clashes
PDB					7	620
SIDEpro	<b>85.01</b>	<b>72.76</b>	<b>0.940</b>	<b>2.06</b>	<b>112</b>	2403
SCWRL4-FRM	84.21	72.21	0.977	19.06	356	3170
SCWRL4-RRM	82.83	69.79	1.030	10.64	172	<b>2117</b>



Table 2.12: CASP9 Dataset Residue Specific Accuracy Results.

AA Type	AA Count	$\chi_1$ (%)		$\chi_{1+2}$ (%)		RMSD (Å)	
		SIDEpro	SCWRL4	SIDEpro	SCWRL4	SIDEpro	SCWRL4
ARG	1118	78.2	<b>78.8</b>	64.3	<b>66.2</b>	<b>2.19</b>	2.24
ASN	901	<b>86.0</b>	83.0	<b>63.3</b>	58.0	<b>0.98</b>	1.07
ASP	1396	<b>85.9</b>	85.5	73.1	<b>73.6</b>	0.81	<b>0.81</b>
CYS	249	89.2	<b>90.8</b>			0.52	<b>0.48</b>
GLN	756	<b>82.7</b>	80.0	<b>65.5</b>	60.3	<b>1.51</b>	1.64
GLU	1488	<b>69.3</b>	69.2	<b>55.4</b>	54.4	<b>1.51</b>	1.53
HIS	548	86.3	<b>87.0</b>	53.3	<b>54.9</b>	1.34	<b>1.31</b>
ILE	1341	<b>94.6</b>	94.6	79.6	<b>81.4</b>	0.53	<b>0.51</b>
LEU	2064	<b>92.0</b>	90.1	<b>83.9</b>	82.7	<b>0.63</b>	0.67
LYS	1172	<b>78.6</b>	77.9	<b>64.9</b>	63.7	<b>1.61</b>	1.66
MET	72	<b>79.2</b>	77.8	<b>76.4</b>	68.1	<b>1.06</b>	1.20
PHE	866	<b>95.6</b>	94.2	<b>91.3</b>	91.2	<b>0.67</b>	0.77
PRO	931	83.9	<b>86.8</b>	75.4	<b>77.8</b>	0.30	<b>0.28</b>
SER	1340	<b>72.6</b>	68.6			<b>0.77</b>	0.85
THR	1105	<b>87.4</b>	86.2			<b>0.48</b>	0.50
TRP	302	<b>92.4</b>	92.1	<b>82.5</b>	76.8	<b>1.18</b>	1.42
TYR	821	<b>94.3</b>	93.8	90.7	<b>91.4</b>	<b>0.86</b>	0.91
VAL	1415	89.0	<b>89.1</b>			0.44	<b>0.43</b>
<b>All</b>	<b>17885</b>	<b>85.01</b>	84.21	<b>72.76</b>	72.21	<b>0.940</b>	0.977

Table 2.13: COMPLEXES Dataset Summary Results: 7 Protein Complexes with 91 Chains, 30734 Residues.

Method	$\chi_1$ (%)	$\chi_{1+2}$ (%)	RMSD (Å)	Time(s)	Severe Clashes	Moderate Clashes
PDB					37	2349
SIDEpro	<b>79.73</b>	<b>64.19</b>	<b>1.109</b>	<b>26.9</b>	<b>253</b>	6114
SCWRL4-FRM	77.99	62.57	1.184	583.9	972	7493
SCWRL4-RRM	76.87	59.87	1.237	409.4	540	<b>5490</b>

Table 2.14: RIBOSOME Dataset Summary Results: 21 Protein Chains, 2001 Residues.

Method	$\chi_1$ (%)	$\chi_{1+2}$ (%)	RMSD (Å)	Time(s)	Severe Clashes	Moderate Clashes
<i>Without RNA</i>						
PDB					0	370
SIDEpro	<b>71.50</b>	<b>53.60</b>	<b>1.504</b>	<b>11.8</b>	<b>13</b>	386
SCWRL4-FRM	70.61	52.37	1.559	180.0	54	433
SCWRL4-RRM	69.57	50.31	1.607	102.1	32	<b>309</b>
<i>With RNA</i>						
SIDEpro	<b>74.30</b>	<b>56.00</b>	<b>1.423</b>	<b>13.7</b>	<b>14</b>	371
SCWRL4-FRM	72.31	54.49	1.480	4020.7	65	477
SCWRL4-RRM	71.46	52.81	1.524	796.2	39	<b>340</b>

Table 2.15: AA Specific Energy vs Generic Energy Tested on Standard Amino Acids

AA Type	AA Specific			Generic			P value
	RMSD	$\chi_1$	$\chi_{1+2}$	RMSD	$\chi_1$	$\chi_{1+2}$	
ARG	<b>2.21</b>	<b>78.7</b>	<b>64.8</b>	2.23	77.9	64.6	<b>0.1019</b>
ASN	<b>1.04</b>	<b>85.4</b>	<b>62.5</b>	1.08	83.9	60.9	<b>0.0002</b>
ASP	<b>0.80</b>	<b>84.9</b>	<b>76.9</b>	0.82	84.3	75.5	<b>0.0272</b>
CYS	0.49	90.0		<b>0.47</b>	<b>90.6</b>		0.1755
GLN	1.69	77.3	58.7	1.69	<b>77.4</b>	58.5	0.8869
GLU	<b>1.46</b>	<b>74.1</b>	<b>58.0</b>	1.48	73.7	57.4	<b>0.0178</b>
HIS	1.31	88.3	<b>54.8</b>	<b>1.29</b>	<b>89.4</b>	53.2	0.3103
ILE	<b>0.44</b>	<b>95.7</b>	<b>84.5</b>	0.46	95.4	82.7	<b>0.0002</b>
LEU	<b>0.53</b>	<b>93.9</b>	<b>87.6</b>	0.55	93.4	86.8	<b>0.0001</b>
LYS	<b>1.63</b>	<b>79.5</b>	<b>66.2</b>	1.69	77.9	64.9	<b>&lt;0.0001</b>
MET	<b>1.09</b>	<b>85.7</b>	<b>77.2</b>	1.11	85.1	75.8	0.1881
PHE	0.67	95.0	92.9	<b>0.66</b>	<b>95.3</b>	<b>93.0</b>	0.1979
PRO	<b>0.26</b>	84.7	<b>81.0</b>	0.30	<b>85.3</b>	79.9	<b>&lt;0.0001</b>
SER	<b>0.75</b>	<b>73.2</b>		0.78	71.5		<b>0.0004</b>
THR	<b>0.39</b>	<b>90.7</b>		0.40	90.1		<b>0.0239</b>
TRP	<b>1.09</b>	92.7	<b>84.4</b>	1.14	92.7	82.7	0.1724
TYR	0.85	94.0	91.4	<b>0.83</b>	<b>94.5</b>	<b>91.6</b>	0.2292
VAL	<b>0.32</b>	<b>93.1</b>		0.34	92.5		<b>0.0057</b>
<b>All</b>	<b>0.91</b>	<b>86.2</b>	<b>74.7</b>	0.93	85.7	73.8	<b>&lt;0.0001</b>

Table 2.16: PTM Specific vs Generic Energy for Frequent PTMs

AA Type	# of Pairs	# of NNs	# of Pairs # of NNs	RMSD		<i>p</i> value
				Specific	Generic	
ABA	6807	5	1361	<b>0.99</b>	1.1433	0.512
CME	408706	15	27247	<b>2.83</b>	2.9194	0.650
CSD	108032	10	10803	<b>1.46</b>	1.598	0.606
CSO	173731	10	17373	<b>1.17</b>	1.18584	0.709
HYP	54104	10	5410	<b>1.05</b>	1.2102	0.411
KCX	1392310	15	92821	<b>1.72</b>	1.79402	0.745
LLP	6902687	15	460179	<b>4.08</b>	5.084	<b>0.002</b>
M3L	309919	10	30992	<b>1.71</b>	1.903	0.195
MLY	1654694	10	165469	<b>2.06</b>	2.1562	<b>0.128</b>
MSE	8102641	10	810264	<b>1.06</b>	1.0818	<b>0.105</b>
OCS	45379	10	4538	0.89	<b>0.8572</b>	0.491
PTR	926740	10	92674	<b>2.02</b>	2.2858	<b>0.143</b>
SEP	121541	10	12154	1.77	<b>1.7664</b>	0.993
TPO	102497	10	10250	<b>1.43</b>	1.4888	0.255

Table 2.17: Accuracy for Frequent PTMs and their Precursor Amino Acid

AA Type	Count	RMSD(Å)		$\chi_1$ (%)	$\chi_{1+2}$ (%)
		Precursor	All		
<b>ALA</b>					
ABA	12	0.99	0.99	61.5	
<b>CYS</b>	<b>1001</b>	<b>0.49</b>		<b>90.0</b>	
CME	16.4	0.86	2.83	75.4	48.0
CSD	16.6	0.56	1.46	91.7	44.0
CSO	44.8	0.64	1.17	88.0	57.1
OCS	17.2	0.60	0.89	86.4	81.6
<b>LYS</b>	<b>3901</b>	<b>1.63</b>		<b>79.5</b>	<b>66.2</b>
KCX	14	1.20	1.72	91.3	66.3
LLP	29.4	1.82	4.08	85.9	36.4
M3L	10	1.31	1.71	82.2	70.6
MLY	51.8	1.43	2.06	76.6	64.6
<b>TYR</b>	<b>2346</b>	<b>0.85</b>		<b>94.0</b>	<b>91.4</b>
PTR	13.8	1.24	2.02	88.1	79.1
<b>SER</b>	<b>4107</b>	<b>0.75</b>		<b>73.2</b>	
SEP	17.2	0.88	1.77	68.0	39.2
<b>THR</b>	<b>3790</b>	<b>0.39</b>		<b>90.7</b>	
TPO	10.4	0.91	1.43	71.9	66.3
<b>MET</b>	<b>1410</b>	<b>1.09</b>		<b>85.7</b>	<b>77.2</b>
MSE	742.6	1.06	1.06	88.0	80.5
<b>PRO</b>	<b>3233</b>	<b>0.26</b>		<b>84.7</b>	<b>81.0</b>
HYP	45.8	0.87	1.05	78.7	67.1
PCA	15.2	0.43	0.48	100	100

Table 2.18: Accuracy of NSA Method on FPTM Set

AA Type	Count	True Structure		COSMOS	
		$\chi_1(\%)$	$\chi_{1+2}(\%)$	$\chi_1(\%)$	$\chi_{1+2}(\%)$
ABA	60	<b>60</b>		55	
CME	82	86.59	<b>86.59</b>	<b>89.02</b>	3.659
CSD	83	<b>86.75</b>	<b>86.75</b>	71.08	12.05
CSO	224	<b>85.2</b>	<b>84.75</b>	81.25	8.482
HYP	229	<b>100</b>	<b>100</b>	95.2	29.26
KCX	70	<b>98.57</b>	<b>98.57</b>	74.29	47.14
LLP	147	<b>89.12</b>	<b>89.12</b>	87.76	42.86
M3L	50	<b>80</b>	<b>80</b>	64	56
MLY	259	<b>83.01</b>	<b>83.01</b>	64.48	52.12
MSE	3713	<b>85.38</b>	<b>85.29</b>	82.01	50.9
OCS	86	<b>91.86</b>	<b>91.86</b>	86.05	74.42
PCA	76	<b>100</b>	<b>100</b>	56.58	55.26
PTR	69	<b>82.61</b>	<b>82.61</b>	69.57	60.87
SEP	86	65.12	<b>65.12</b>	<b>66.28</b>	25.58
TPO	52	70.59	<b>70.59</b>	<b>75</b>	63.46

Table 2.19: RMSD( $\text{\AA}$ ) of NSA Method on FPTM Set

AA Type	Count	True Structure		COSMOS	
		precursor	all	precursor	all
ABA	60	<b>1.054</b>	<b>1.054</b>	1.162	1.162
CME	82	<b>0.6276</b>	<b>0.3462</b>	0.6648	4.783
CSD	83	<b>0.6829</b>	<b>0.782</b>	1.17	2.387
CSO	224	<b>0.6782</b>	<b>0.6404</b>	0.8186	1.892
HYP	229	<b>0.1067</b>	<b>0.1331</b>	0.5366	0.5366
KCX	70	<b>0.3971</b>	<b>0.5761</b>	2.151	3.377
LLP	147	<b>1.056</b>	<b>2.029</b>	1.919	6.709
M3L	50	<b>1.125</b>	<b>1.407</b>	2.059	2.535
MLY	259	<b>1.044</b>	<b>1.259</b>	2.098	2.777
MSE	3713	<b>0.367</b>	<b>0.367</b>	2.023	2.023
OCS	86	<b>0.5527</b>	<b>0.7119</b>	0.785	1.096
PCA	76	<b>0.08948</b>	<b>0.08601</b>	0.7811	0.8356
PTR	69	<b>1.43</b>	<b>2.019</b>	2.215	3.553
SEP	86	<b>0.8974</b>	<b>1.411</b>	0.9638	2.054
TPO	52	0.9129	<b>1.397</b>	<b>0.9065</b>	1.793

Table 2.20: Accuracy of NSA Method on NSA (non-FPTM) Test Set

Default structures	RMSD(Å)	$\chi_1$ (%)
Single Predicted Structure	3.54	56.39
10 Predicted Structures	3.08	65.30
True Structure	<b>1.75</b>	<b>66.63</b>

Table 2.21: Statistical Significance (p-values) of SIDEpro Improvement with Respect to SCWRL4. Results significant at  $p < 0.001$  are shown in bold.

Dataset	AA Count	$\chi_1$	$\chi_{1+2}$	RMSD
<i>SIDEpro wrt SCWRL4-FRM</i>				
SCWRL4	58265	<b>3.1e-08</b>	<b>9.8e-05</b>	<b>&lt;1.0e-10</b>
CASP9	17885	<b>7.1e-04</b>	5.1e-02	<b>3.4e-08</b>
COMPLEXES	30734	<b>&lt;1.0e-10</b>	<b>1.5e-09</b>	<b>&lt;1.0e-10</b>
RIBOSOME	2001	1.7e-01	1.4e-01	1.4e-02
RIBOSOME + rna	2001	1.5e-02	9.3e-02	1.4e-02
<i>SIDEpro wrt SCWRL4-RRM</i>				
SCWRL4	58265	<b>&lt;1.0e-10</b>	<b>&lt;1.0e-10</b>	<b>&lt;1.0e-10</b>
CASP9	17885	<b>&lt;1.0e-10</b>	<b>&lt;1.0e-10</b>	<b>&lt;1.0e-10</b>
COMPLEXES	30734	<b>&lt;1.0e-10</b>	<b>&lt;1.0e-10</b>	<b>&lt;1.0e-10</b>
RIBOSOME	2001	2.1e-02	2.2e-03	<b>2.1e-05</b>
RIBOSOME + rna	2001	1.3e-03	3.1e-03	<b>4.1e-05</b>

# Chapter 3

## Contact Map Prediction

### 3.1 Introduction

Protein residue-residue contact prediction is the problem of predicting whether any two residues in a protein sequence are spatially close to each other in the folded 3D structure. Contacts occurring between sequentially distant residues, i.e. long-range contacts, impose strong constraints on the 3D structure of a protein and are particularly important for structural analyses, understanding the folding process, and predicting the 3D structure. Even a small set of correctly predicted long-range contacts can be useful for improving ab-initio structure prediction for proteins without known templates [81].

The performance of many contact predictors has been assessed every two years during the CASP experiments, since CASP2 in 1996. Unfortunately, the  $\sim 20\%$  accuracy for long-range contacts, routinely reported at CASP for the best predictors [26, 46], suggests that contact prediction is not yet accurate enough to be systematically useful for ab-initio protein structure prediction or engineering.

In broad terms, there are four main approaches for residue-residue contact prediction. *Machine learning* approaches use methods such as neural networks [28, 70, 76], recursive neural networks [17, 83], support vector machines [17], and hidden Markov models [10] to learn how to predict contact probabilities, from a training set of experimentally determined protein structures. Inputs to these approaches typically include predicted secondary structure, predicted solvent accessibility, as well as evolutionary information in the form of profiles. *Template-based* approaches use homology or threading methods to identify structurally similar templates from which residue-residue contacts are then inferred [79, 57]. *Correlated mutations* approaches apply statistical measures, such as Pearson correlation [31, 65] and mutual information [24, 14], to multiple alignments in order to identify pairs of residues that co-evolve and thus are likely to be in contact. Recently, a new elegant mutual information-based measure for correlated mutations, PSICOV, has been proposed in [37] and used for fold recognition [80]. While this method has been reported to yield significant accuracy improvements, its performance is very dependent on the availability and quality of multiple alignments. Finally, *3D model-based* approaches rely on predicted 3D structures for deriving distance constraints through a consensus strategy. Although 3D model-based approaches have been reported to be the most accurate at CASP [46], in practice their applicability remain somewhat limited since the main goal of contact prediction is to improve ab-initio structure prediction, and not the converse.

Here we introduce several new ideas for contact prediction using primarily a multi-stage machine learning approach, with increasingly refined levels of resolution. First, we predict coarse contact maps corresponding to contacts between secondary structure elements. By itself, the idea of coarse contact maps is not new and several useful methods have been developed [82, 17, 69]. Yet none of these approaches has been able to convincingly demonstrate that coarse prediction is useful for residue-residue contact prediction. Here we both refine the previous coarse prediction methods, in part by extending the notion of coarse contact beyond a simple binary value to include information about orientation (parallel vs antipar-



allel) between contacting segments, and demonstrate that coarse-grained prediction can be used to improve fine-grained prediction of contact maps. Second, we use a novel energy based neural network approach to refine the prediction of the alignment and orientation of contacting secondary structure elements and predict residue-residue contact probabilities for residues in contacting pairs of alpha-helices or beta-strands. Finally, we introduce a deep neural network architecture in the form of a deep stack of neural networks, with the same topology but different parameters, to predict all the residue-residue contact probabilities by integrating information both spatially and temporally. Spatial integration refers to the idea that contacts are spatially correlated, for instance long-range contacts often include other long-range contacts in their neighborhood. Temporal integration refers to the idea that protein folding is not an instantaneous physical process. While the stack is not necessarily meant to mimic the actual physical process, the stack is used to organize the prediction in such a way that each level in the stack is meant to refine the prediction produced by the previous level. Inputs at a given level of the stack include both information coming from the previous level in the stack as well as static information produced by the previous coarse prediction stages, as well as predicted secondary structure and solvent accessibility, and evolutionary profiles. Thus these dynamic and static inputs are used to iteratively refine the contact prediction. We next describe these methods in detail together with the data used for rigorous training and assessment results.

## 3.2 Materials and Methods

### 3.2.1 Contact definition and evaluation criteria

We adopted the same intramolecular contact definition and the same evaluation criteria as in the most recent CASP experiments. Two residues are defined to be in contact if the Euclidean distance between their  $C_\beta$  atoms ( $C_\alpha$  for Glycines) is lower than  $8\text{\AA}$ . Three

distinct classes of contacts are defined, depending on the linear sequence separation between the residues: (1) long-range contacts, with separation  $\geq 24$  residues; (2) medium-range contacts, with separation between 12 and 23 residues; and (3) short-range contacts, with separation between 6 and 11 residues. Contacts between residues separated by less than 6 residues are dense and can be easily predicted from the secondary structure. Conversely, the sparse long-range contacts are the most informative and also the most difficult to predict. Thus, as in the CASP experiments, we focus primarily on long-range contact prediction.

The performance is evaluated using two main measures: the accuracy ( $Acc$ ) and the distance distribution ( $X_d$ ). The accuracy is defined as the fraction of correctly predicted contacts with respect to the total number of contacts evaluated:

$$Acc = TP/(TP+FP),$$

where TP and FP are the true positive and false positive predicted contacts, respectively. The distance distribution score measures the weighted harmonic average difference between the predicted contacts distance distribution and the all-pairs distance distribution. The  $X_d$  is defined by

$$X_d = \sum_{i=1}^{15} \frac{(Pp_i - Pa_i)}{i},$$

where  $Pp_i$  is the fraction of predicted pairs whose distance is in the bin  $d_i = [4(i - 1), 4i]$  and  $Pa_i$  is the fraction of all pair of targets in the bin  $d_i$ . The higher  $X_d$  is, the better the performance (a random predictor corresponds to  $X_d = 0$ ). Contact predictors usually assign a probability score to every possible pair of residues or to a subset of the possible pairs. The  $Acc$  and  $X_d$  measures are computed for the sets of  $L/5$ ,  $L/10$  and 5 top scored predicted pairs, where  $L$  is the length of the domain sequence. While predictions are evaluated on all three sets, the most widely used performance measure is  $Acc$  for  $L/5$  pairs and sequence separation  $\geq 24$ .

### 3.2.2 Training and test sets

The training set is derived from the ASTRAL database [16]. We extract from the ASTRAL release 1.73 the (precompiled) set of protein domains with less than 20% pairwise sequence identity, removing domains of length less than 50 residues, domains with multiple 3D structures, as well as non-contiguous domains (including those with missing backbone atoms). We further filter this list by selecting just one representative domain—the shortest one—per SCOP family [60], ending up with a final set of 2,356 structures. For cross-validation purposes, this set is then partitioned into 10 disjoint groups of roughly the same size and average domain lengths, so that no domains from two distinct groups belong to the same SCOP fold. In this way, training and validation sets share neither sequence nor structural similarities.

For performance assessment, a non-redundant test set is derived from ASTRAL release 1.75, by selecting all the new folds, with respect to version 1.73, belonging to the main SCOP classes (all-alpha, all-beta, alpha/beta and alpha + beta). From this set (256 new folds and 287 new families), we remove all the domains of length  $<50$  residues and those with  $<L/5$  long-range contacts (239 new folds and 268 new families). Redundancy is filtered out by clustering each group of domains belonging to the same SCOP family at 40% of sequence similarity. The final set of 364 domains contains at least one representative for each one of the 268 new families. A BLAST [1] search with E-value cutoff 0.01 of the test domain sequences against the set of training domain sequences returns no hit.

For comparison with the current state-of-art contact predictors, the performance is tested on the template-based/free-modeling (TBM/FM) domain targets used in the last two CASP experiments, CASP8 [26] and CASP9 [58] for contact prediction assessment. Note that ASTRAL 1.73 was released in 2007, before the CASP8 experiment held in 2008, thus no structural similarities exist between the domains in the training set and those from CASP8 (12 domains) and CASP9 (28 domains). An additional test is performed with BLAST [1] to

detect sequence similarities between the CASP and the training target sequences. A BLAST search with an E-value cutoff of 0.01 returns no similar domain pairs. The predictions for the groups participating at the CASP8 and CASP9 meetings are obtained from the CASP website<sup>1</sup>. As in CASP, performance is assessed here only at the domain level, although predictions are available for the entire protein targets. To simplify the comparison, we select only those groups that submitted a prediction for all the targets in the respective CASP8 and CASP9 sets. Furthermore, we considered all the domain targets for each group, regardless of the number of predicted contacts per domain. CASP assessors typically exclude from the analysis the results of a predictor on any domain where the number of predicted contacts is not high enough. This filtering step is not used here since it does not affect the performance of the top-scoring predictors.

### 3.2.3 Coarse contact and orientation prediction (BRNN)

We use two-dimensional bidirectional recurrent neural networks (2D-BRNNs) [17] to predict coarse contact probabilities and orientations between secondary structure elements. Specifically, ignoring for robustness coils, short strands ( $\geq 3$  residues), and short helices ( $\geq 6$  residues), we predict the probability of whether two elements are in parallel contact, antiparallel contact, or no-contact. The distance between two secondary structure elements is defined to be the minimum Euclidean distance among all the possible pairs of  $C_\alpha$  atoms, one from each element. A pair of elements is defined to be in contact if and only if their distance is less than  $8\text{\AA}$ . The orientation angle of two elements is defined as the angle between their orientation vectors. The orientation vector is computed by joining the centers of gravity ( $C_\alpha$  coordinates) of the first and second half of the element. Two elements are in parallel contact if their distance is less than  $8\text{\AA}$  and their orientation angle is less than 90 degrees, anti-parallel contact if their distance is less than  $8\text{\AA}$  and their orientation angle is greater

---

<sup>1</sup><http://predictioncenter.org/>

than 90 degrees, and no-contact if their distance is larger than 8Å.

For each pair  $S_n$  and  $S_m$  of secondary structure elements, the output of the 2D-BRNN is a probability vector corresponding to the probability of parallel contact, anti-parallel contact, or no-contact. The input of the 2D-BRNN for the pair  $S_n, S_m$  consists of two feature vectors  $F_n$  and  $F_m$ , as well as the number of elements between  $S_n$  and  $S_m$ . The feature vector  $F_n$  for segment  $S_n$  has the following components.

1. Three vectors (20 entries each), representing the average amino acid distribution computed over the profiles of  $S_{n-1}$ ,  $S_n$  and  $S_{n+1}$ .
2. The lengths (3 entries) in residues of  $S_{n-1}$ ,  $S_n$ , and  $S_{n+1}$ .
3. The lengths (2 entries) in residues of the intervals between  $S_{n-1}$  and  $S_n$ , and  $S_n$  and  $S_{n+1}$ . These intervals correspond to the sum of the lengths of the coils and short elements that are ignored between the elements under consideration. This length is 0 for adjacent elements (Figure 3.1).
4. A vector of flags (4 binary entries) to identify the first, second, second-to-last, and last elements in the sequence.
5. Two vectors (20 entries each) containing the average amino acid distribution for alternate even- and odd-numbered columns in the profile of  $S_n$ . Specifically, if  $S_n$  consists of residues  $s_1, s_2, s_3, \dots$ , the first vector contains the average sequence profile over residues  $(s_1, s_3, s_5, \dots)$  and the second vector over  $(s_2, s_4, s_6, \dots)$ . This feature is designed explicitly for strands, since these two sets of positions tend to have similar properties when the two strands are paired in a beta-sheet.

The 2D-BRNN is trained using 10-fold cross-validation.

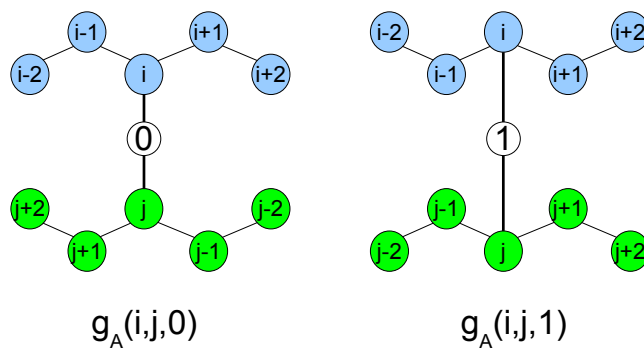
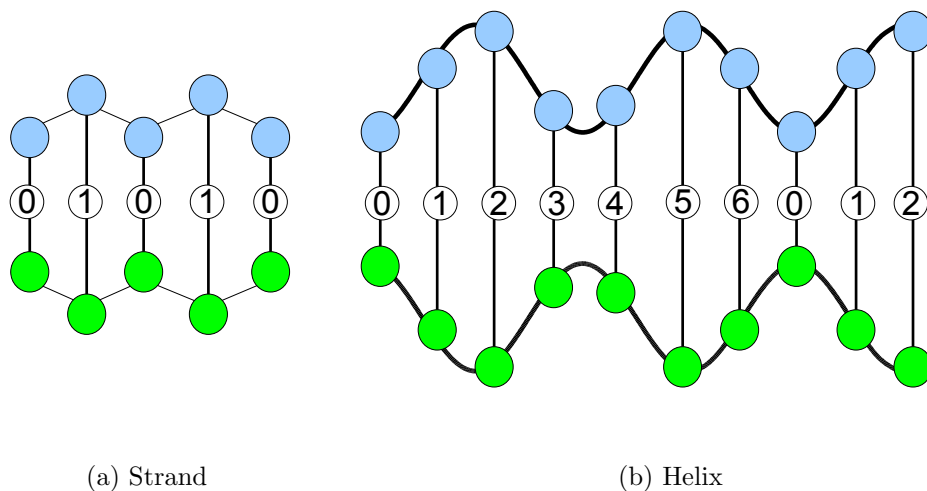


### 3.2.4 Element alignment prediction (Energy)

We use an energy-based method [61] to assign energies then probabilities to the alignment between contacting secondary structure elements and derive approximate probabilities of contact for their residue pairs. This approach is used only for helix-helix and strand-strand contacting elements, since these are by far the most frequent among well defined secondary structure elements (i.e. strand-helix contacts are relatively rare). Furthermore, it is generally hard to align strand and helix elements at the residue-level because helices are more compact when compared to strands.

Alignments between secondary structure elements are described by two components: the relative shift and the phase. The relative shift is an integer representing how the residues in the first element are shifted with respect to the second element. For instance, the shift between two strands of length 5 can have any integer value from 0 to 9. The phase is an integer assigned to pairs of residues, one from each contacting element, which is meant to capture in approximate fashion the periodic component of strand-strand and helix-helix contacts with some partial correlation to physical distance. Since the side-chains of contacting strands are alternatively distributed on each side of the corresponding beta-sheet, and alpha-helices make approximately two turns every seven residues, it is reasonable to view strands and helices as periodic structures with periods 2 and 7 respectively. The phase value is assigned periodically by starting from the two residues with the closest  $C_\alpha$ s and moving away from it in both directions. For strand-strand contacts, the phase values alternate between 0 and 1, while for helix-helix contacts, the phase values cycle periodically from 0 to 6 (Figure 3.2(a) and 3.2(b)).

Given a pair of contacting elements  $S_n$  and  $S_m$ , we need to evaluate the energy of all the possible alignments obtained by shifting  $S_n$  over  $S_m$  (which is kept fixed), such that at least one residue in  $S_n$  is paired with one residue in  $S_m$ . If  $|S_n| = k_n + 1$  and  $|S_m| = k_m + 1$  are the



(c) Phase type and corresponding energy term for anti-parallel strands

Figure 3.2: (a) Phase values (0-1) for pairs of residues in contacting strands. (b) Phase values (0-6) for pairs of residues in contacting helices. (c) Phase values for anti-parallel strands and corresponding energy terms  $g_A(i, j, k)$ .



lengths of  $S_n$  and  $S_m$ , there are exactly  $k_n+k_m+1$  possible shifts numbered  $a = 0, 1, \dots, k_n+k_m$ . Each one of these shifts can be in  $O$  different phases numbered  $\theta = 0, \dots, O - 1$ , with  $O = 2$  for strands and  $O = 7$  for helices. Thus we need to evaluate the energy of  $O \cdot (k_n + k_m + 1)$  alignments. Assume that the segment  $S_n$  consists of residues  $i, i + 1, \dots, i + k_n$  and  $S_m$  of residues  $j, j + 1, \dots, j + k_m$ . Then, the energy for the  $a$ -th shift with phase  $\theta$  of segment  $S_n$  versus segment  $S_m$  is given by

$$E_P(a, \theta) = \sum_{k=0}^{k_m} g_P(i - k_m + a + k, j + k, (\theta + k) \bmod O) \quad (3.1)$$

$$E_A(a, \theta) = \sum_{k=0}^{k_m} g_A(i - k_m + a + k, j + k_m - k, (\theta + k) \bmod O) \quad (3.2)$$

where the function  $g_P(i, j, k)$  (resp.  $g_A(i, j, k)$ ) returns the estimated energy for the residue pair  $i, j$ , under the assumption that  $S_n$  and  $S_m$  are parallel contacting (resp. anti-parallel contacting) and that the phase of  $i, j$  is  $k$  (Figure 3.2(c)). As a manageable example, Figure 3.3(a) shows all the alignment positions and the corresponding energies for two anti-parallel strands of hypothetical length 3. The alignment energies  $E_P(a, \theta)$  and  $E_A(a, \theta)$  are normalized into probabilities by

$$P_P(a, \theta) = \frac{e^{-K \cdot E_P(a, \theta)}}{\sum_{j=0}^{k_m+k_n} \sum_{k=0}^{O-1} e^{-K \cdot E_P(j, k)}} \quad (3.3)$$

$$P_A(a, \theta) = \frac{e^{-K \cdot E_A(a, \theta)}}{\sum_{j=0}^{k_m+k_n} \sum_{k=0}^{O-1} e^{-K \cdot E_A(j, k)}} \quad (3.4)$$

where  $K$  is a fixed constant.

In order to compute the alignment energies (3.1) and (3.2) and the corresponding normalized probabilities (3.3) and (3.4), we need to define the residue-residue energy functions  $g_P(i, j, k)$  and  $g_A(i, j, k)$ . We model these functions by using two-layer feedforward NNs. There are four NNs: two for the strand-strand parallel and anti-parallel cases, and two for the helix-

helix parallel and anti-parallel cases. In all four cases, the NN input simply encodes the two sequence profile vectors (20 entries each) for the residue pair  $(i, j)$ . The output size of the NNS is  $O = 2$  for the strand-related predictors and  $O = 7$  for the helix-related predictors and represents the phases. The function  $g_A(i, j, k)$  thus represents the  $k$ -th output of the (anti-parallel) NN for the residue pair  $(i, j)$ . The network weights for the anti-parallel case are trained by gradient-descent minimization of the log-likelihood objective function

$$\mathbf{E}_A = - \sum_{i=1}^n \log P_A(\hat{a}_i, \hat{\theta}_i) \tag{3.5}$$

where  $n$  is the number of anti-parallel contacting element pairs used in training and  $\hat{a}_i, \hat{\theta}_i$  are the true shift and phase for the  $i$ -th example (the objective function is similar for the parallel case). Thus, we can train the NN weights by gradient descent, back-propagating the partial derivatives:

$$\frac{\partial \mathbf{E}_A}{\partial E_A(a_i, \theta_i)} = \begin{cases} -K \cdot (P_A(a_i, \theta_i) - 1), & (a_i, \theta_i) = (\hat{a}_i, \hat{\theta}_i) \\ -K \cdot P_A(a_i, \theta_i), & \text{otherwise} \end{cases} \tag{3.6}$$

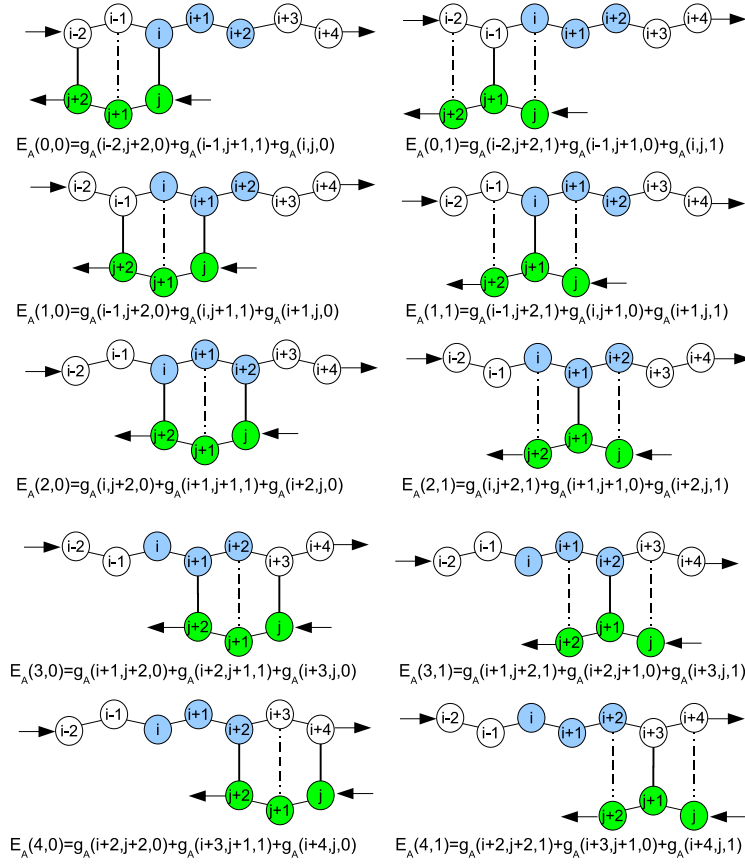
The four alignment predictors are also trained using 10-fold cross-validation on the data described in Section 3.2.2.

The alignment probabilities provide an estimation of the possible spatial arrangement of two secondary structure elements. These probabilities can easily be mapped to residue-residue contact probabilities. The mapping is obtained by choosing the probability score of the unique alignment in which the two residues are paired together and are close (i.e. their phase is 0). For instance, assume that  $i$  and  $j$  belong to two anti-parallel elements  $S_n$  and  $S_m$ . Then, there is a unique shift  $a$  of  $S_n$  over  $S_m$  in which  $i$  and  $j$  are paired together. For this shift, there is a unique overall phase  $0 \leq \theta < O$  such that  $i$  and  $j$  are given phase 0.

Then, the probability  $P_A(a, \theta)$  represents the probability of contact for the pair  $(i, j)$  (Figure 3.3(b)).

### 3.2.5 Residue-residue contact prediction (Deep NN)

The deep neural network architecture for residue-residue contact prediction consists of a three dimensional stack of neural networks  $\text{NN}_{ij}^k$ . Each network  $\text{NN}_{ij}^k$  in the stack is a standard three-layer feed-forward network trainable by back propagation, and all the networks share the same topology: same input size, same hidden layer size, with one single output, which represents the residue-residue contact probability computed at position  $i, j$  and level  $k$ . Thus  $i$  and  $j$  are spatial indexes over the contact map, whereas  $k$  is a “temporal” index. Each layer  $k$  of NNs in the stack produces a contact map prediction, which is then refined in the subsequent layers. The range of  $k$  is determined during the training phase, as described below. Each  $\text{NN}_{ij}^k$  has two different kinds of input features: purely spatial features and temporal features. For fixed  $i$  and  $j$ , the purely spatial features are identical for all the  $\text{NN}_{ij}^k$  as  $k$  varies and consist of typical features used in contact map prediction. The temporal input features for  $\text{NN}_{ij}^{k+1}$  consist of the predicted contact map around  $i$  and  $j$  at the previous level of the stack, i.e. the outputs of the networks  $\text{NN}_{rs}^k$ , where  $r, s$  ranges over a “receptive field” neighborhood of  $i, j$ . The receptive fields used in the simulations results are essentially  $15 \times 15$  square patches (Figure 3.5). The integration over time provided by the different levels in the stack corresponds to the intuition that folding is a somewhat organized, non-instantaneous, process which proceeds through successive stages of refinement. The integration over space provided by the receptive fields of the temporal features captures the idea that residue-residue contacts in native protein structures are generally not isolated: a contacting residue pair is very likely to be in the proximity of a different pair of contacting residues. Over 98% of long-range contacting residues are in close proximity of another contact, compared to 30% for non-contacting pairs. Furthermore, over 60% of contacting pairs are in the proximity of

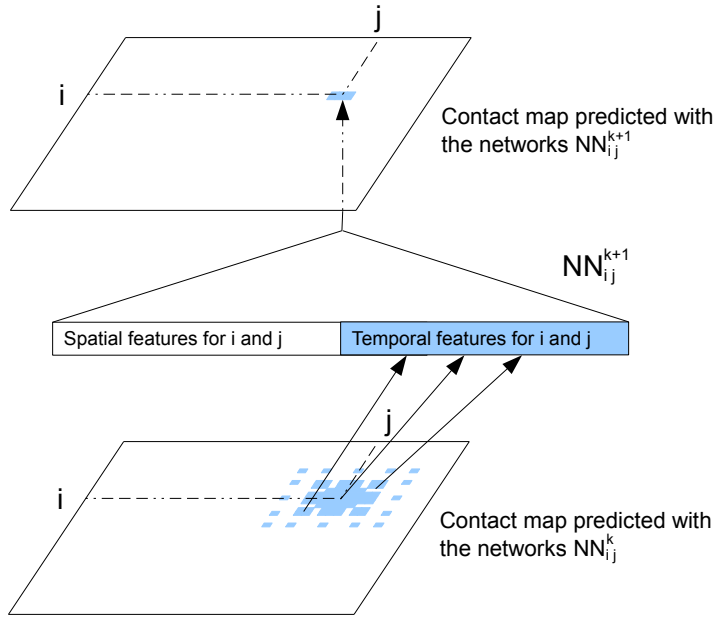


(a) Alignments and corresponding energies

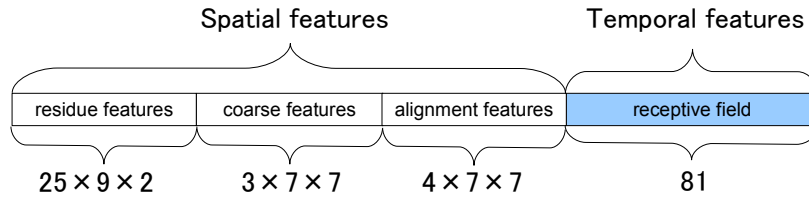
	j	j+1	j+2
i	$P_A(0,0)$	$P_A(1,1)$	$P_A(2,0)$
i+1	$P_A(1,0)$	$P_A(2,1)$	$P_A(3,0)$
i+2	$P_A(2,0)$	$P_A(3,1)$	$P_A(4,0)$

(b) Mapped contact probabilities

Figure 3.3: (a) Alignments and corresponding energies for two anti-parallel beta-strands of hypothetical length 3. (b) Alignment probabilities mapped to residue-residue contact probabilities for the two elements.



(a) Deep-NN Architecture



(b) NN input features

Figure 3.4: (a) The deep-NN architecture consists of a 3D stack of neural networks  $\text{NN}_{ij}^k$  with identical architecture, but different weights. When  $i$  and  $y$  vary, the outputs of the  $\text{NN}_{ij}^k$  correspond to the predicted contact map at level  $k$  of the stack. A neural network  $\text{NN}_{ij}^{k+1}$  purely spatial input features that depend only on  $i$  and  $j$  and are identical at all levels of the stack, and temporal input features associated with the contact probabilities predicted in the previous layer over a receptive field neighborhood of  $ij$ . (b) Input feature vector of each  $\text{NN}_{ij}^k$ .

at least 10 different contacts, compared to 2.5% for non-contacting pairs (Figure 3.6). In other words, for a residue pair  $(i, j)$ , the higher the number of its neighboring contact pairs, the higher the probability that  $i$  and  $j$  are in contact. Most previous machine learning-based contact predictors learn the contact probabilities of residue pairs independently of the contact probabilities in their neighborhoods. Thus one of the aims of the deep-NN architecture is to leverage this important information during the learning phase. Note that even if the individual contact predictions at a given stage are inaccurate, the contact probabilities can still provide a rough estimate of the number of contacts in a given neighborhood.

There are three types of purely spatial input features: residue-residue features coarse features, and alignment features. **Residue-residue features** encodes three kinds of information (for a total of 25 values): evolutionary information (20 values, one for each amino acid type), predicted secondary structure (3 binary values,  $\beta$ -sheet,  $\alpha$ -helix, or coil) and predicted solvent accessibility (2 binary values, buried or exposed). The evolutionary information is encoded in the standard way, as residue frequency profiles extracted from multiple sequence alignments. Frequency profiles are obtained by running PSI-BLAST [2] with E-value cut-off equal to 0.001 and up to ten iterations against NCBI's non-redundant protein sequence database NR. The secondary structure is predicted with SSPRO [68] and the solvent accessibility with ACCPRO [67]. We used two versions of SSPRO and ACCPRO older than 2008, without retraining them. We used two previously published versions of SSpro [68] and ACCpro [67], derived before 2008 [18], without retraining them. The residue-residue features for the pair  $(i, j)$  are included in the network input by taking a fixed-size window centered at each residue. That is, for the pair  $(i, j)$ , the network input includes the residue-residue feature vectors for residues  $i' \in [i - l, i + l]$  and  $j' \in [j - l, j + l]$ , where  $l \geq 0$  is the radius of the window. After some experimentation, we use  $l = 4$  since larger radiuses lead to slower training with no significant performance improvement. **Coarse features** (3 values) contain the predictions obtained with the coarse contact and orientation predictor (see Section 3.2.3). If residues  $i, j$  are in elements  $S_n, S_m$ , the feature vector is setup with the predicted contact

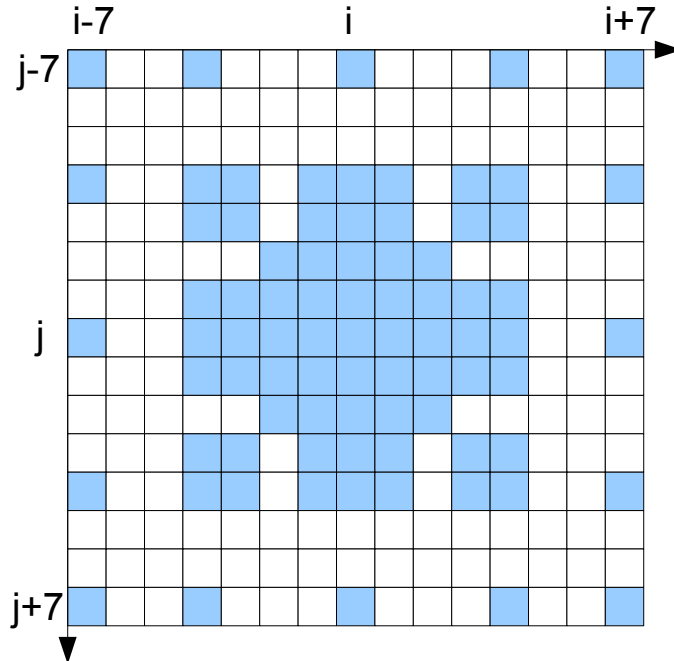


Figure 3.5: Neighborhood of  $(i, j)$ , or receptive field, of the level-dependent inputs. The temporal input feature vector (81 values) contains contact predictions computed in the previous layer of the stack. That is, for a pair of residues  $(i, j)$ , the network-specific feature vector of  $\text{NN}_{ijk+1}$  contains the contact predictions obtained by the networks  $\text{NN}_{rsk}$  for all the pairs  $(r, s)$ , where  $r \in [i - l, \dots, i + l]$ ,  $s \in [j - l, \dots, j + l]$  and  $l \geq 0$  is the “radius” of the neighborhood. We considered different topologies for the receptive pattern and different sizes for the radius parameter. In general, we found that larger radiuses provide better performance. On the other end, larger radiuses imply larger input and thus a longer training time and risk of overfitting. Our best results are obtained using a radius equal to 7 and the receptive field shown in Figure 3.5. The details of this receptive field are derived from the pattern of contacts in native contact maps between helical secondary structures. Since helical structures go around twice every seven residues or so, they tend to have contacts at  $(i \pm 7, j \pm 7)$  when the residue pair  $(i, j)$  is in contact. Furthermore, they also tend to have contacts at  $(i \pm 7, j \pm 4)$  or  $(i \pm 4, j \pm 7)$  because helical structures take a turn every 3-4 residues.

orientation probabilities (parallel, anti-parallel, and non-contact) for  $S_n$  and  $S_m$  (Figure 3.1). If either  $S_n$  or  $S_m$  is an ignored elements (i.e. coil element or short helix/strand element) the three values in the feature vector are set by default to zero. The coarse contact features are included in the network input by taking a fixed-size window (of radius 3) centered at the element pair. **Alignment features** (4 values) contain the predictions obtained with

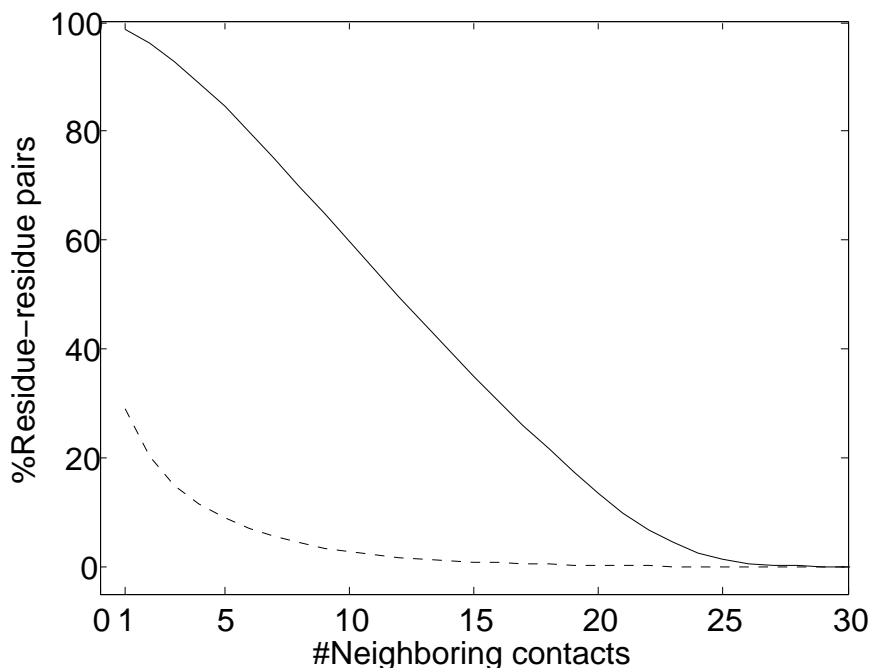


Figure 3.6: Contact proximity distribution for contacting residue pairs (continue line) and non-contacting residue pairs (dotted line), computed over long-range pairs of residues.

the element alignment predictor (see Section 3.2.4). If residues  $i, j$  are in elements  $S_n, S_m$  and  $S_n, S_m$  are both helix elements, the first and second entries of the vector contain the alignment probability score between  $i$  and  $j$  for the cases parallel and anti-parallel contact, respectively. The remaining two entries are set to zero. The encoding is symmetrical for the strand-strand case. If  $S_n$  and  $S_m$  are not both helix or both strand elements, the four entries of the feature vector are set by default to zero. As for the coarse contact features, the alignment features are included in the network input by taking a fixed-size window (of radius 3) centered at the residue-residue pair.



## Deep-NN training.

Training deep multi-layered neural networks is generally hard, since the backpropagated gradient tends to vanish or explode with a high number of layers [47]. Here we use an incremental approach to overcome this problem. The weights of the first level networks,  $NN_{ij}^1$ , are randomly initialized and their temporal input features set to 0. These networks are then trained by on-line back-propagation for one epoch. The weights of  $NN_{ij}^1$  are then used to initialize the weights of  $NN_{ij}^2$  and all the outputs of the  $NN_{ij}^1$  networks on the training set are stored and used to compute the temporal input features of the networks  $NN_{ij}^2$  which are then trained by back-propagation during one epoch. Then the weights of the networks  $NN_{ij}^2$  are used to initialize the weights of the networks  $NN_{ij}^3$  and so forth all the way to the top of the stack. This progressive initialization is critical: initialization with random weights at each level of the stack results in poor performance, from unstable learning to getting stuck in poor local minima. Likewise, more stable training is obtained by using the same training set at each level of the stack, as opposed to randomizing the training data. Thus in practice at each training epoch we append a new neural network to the growing deep-NN architecture, initialize it with the weights of the previous level, and train it by back-propagation using the true contacts as the targets (or softer targets could be derived from folding data). We have experimented with many variations such as growing the stack up to a maximum of 100 networks, or growing it to a smaller depth but then repeating the training procedure through one or more epochs. The approach described earlier in the text provides a good compromise between training time and average cross-validation accuracy. Note that, although a deep-NN with  $n$  levels comprises  $n \times 3$  layers, the number of free training parameters is rather small. Only the parameters of the first level are free, all other parameters are initialized in succession using the parameters from the previous level after one training epoch.

Since the non-contact pairs are considerably more abundant than the contact pairs, a standard approach to deal with unbalanced training set is to rebalance the data. For contact

map prediction, this is often done randomly selecting only 5% of the negative examples, while keeping all the positive examples. In our experiments, we obtain considerable better overall performance by increasing this percentage to 20% (data not shown).

We train 10 different deep-NN predictors by cycling through the 10 training subsets (Section 3.2.2), each time holding one subset for early stopping or validation purposes. Furthermore, we synchronize the early stopping across the 10 deep-NN architectures, so that they all have the same depth  $n$ , retaining the depth providing the best prediction performance ( $n = 71$ , Figure 3.7).

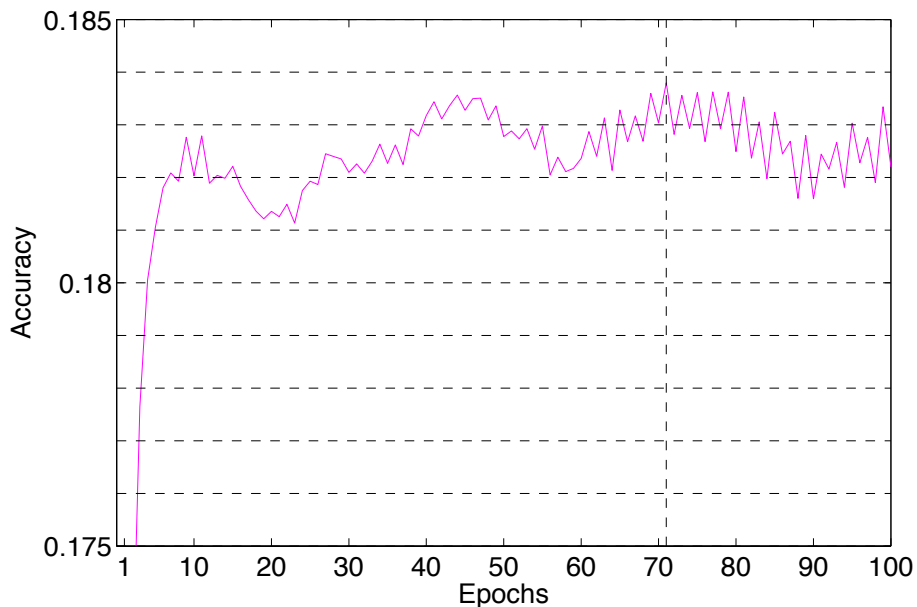


Figure 3.7: Cross validation performance of CMAPpro for the selection of the best network depth. The curve plots the *cross-validation accuracy - standard deviation over the ten validation sets* for the set of true number of long range contacts. The accuracy on the true number of long range contacts shows more variation than the accuracy on the  $L/5$  long range contacts. The highest peak of accuracy is found at depth 71.

Table 3.1: Average performance for the coarse contact and orientation predictor.

Class	$Q_3$	Parallel ( $P$ )		Anti-parallel ( $A$ )		No-contact ( $N$ )	
		$PPV_P$	$TPR_P$	$PPV_A$	$TPR_A$	$PPV_N$	$TPR_N$
All	0.80	0.45	0.15	0.65	0.39	0.82	0.95
All-Alpha	0.73	0.40	0.13	0.63	0.47	0.77	0.92
All-Beta	0.86	0.29	0.06	0.69	0.35	0.88	0.97
Alpha/Beta	0.81	0.57	0.25	0.67	0.36	0.83	0.96
Alpha+Beta	0.79	0.38	0.09	0.64	0.38	0.82	0.95

Parallel contact ( $P$ ), Anti-parallel contact ( $A$ ) and No-contact ( $N$ ) are the three classes considered by the coarse contact and orientation predictor.  $Q_3$  is the percentage of correctly predicted pairs in Equation (3.7),  $PPV_X$  is the Positive Predictive Value on class  $X$  in Equation (3.8) and  $TPR_X$  is the True Positive Rate on class  $X$  in Equation (3.9).

### 3.3 Results and Discussion

#### 3.3.1 Coarse contact and orientation prediction

We evaluate the average classification performance of the coarse contact and orientation predictor on the three classes Parallel contact ( $P$ ), Anti-parallel contact ( $A$ ) and No-contact ( $N$ ) on the 364 test domains (Section 3.2.2). We evaluate the performance using the percentage of correctly predicted pairs

$$Q_3 = \frac{PP + AA + NN}{\sum_X \sum_Y XY} \quad (3.7)$$

the Positive Predictive Value (or precision)

$$PPV_X = (XX)/(AX + PX + NX) \quad (3.8)$$

and the True Positive Rate (or recall)

$$TPR_X = (XX)/(XA + XP + XN) \quad (3.9)$$

where  $XY$  denotes the number of segment pairs in class  $X \in \{P, A, N\}$  predicted to be in class  $Y \in \{P, A, N\}$ . Table 3.1 reports the cross-validation average performance on the full set of protein domains (All) and as a function of the main structural domain classes: All-Alpha (mainly alpha-helices), All-Beta (mainly beta-sheets), Alpha/Beta (alpha-helices and beta-sheets, mainly parallel beta sheets) and Alpha+Beta (alpha-helices and beta-sheets, mainly anti-parallel beta sheets). As shown in Table 3.1, the performance of the coarse predictor on the Parallel (P) class are highly affected by the protein structural domain; in particular, the prediction precision and recall are higher for the Alpha/Beta proteins and are quite low for the All-Beta proteins. Conversely, the performance on the Anti-parallel class (A) are nearly uniform, regardless of the domain structural classification. The anti-parallel contacts appear to be easier to predict than the parallel contacts, even within the Alpha+Beta class. Though not directly comparable (due to a different definition of segment-segment contact), the coarse contact predictor has higher precision and lower recall than the 2D-BRNN developed for the same classification problem in [69].

### 3.3.2 Element alignment prediction

We evaluate the contact prediction performance of the element alignment predictor at the residue level on the (predicted) strand-strand and helix-helix regions of the contact map. We use the same accuracy measure adopted for the evaluation of contact prediction performance on the entire contact map (Section 3.2.1).

Recall that the element alignment predictor can be used to derive approximate probabilities of contacts for residue pairs in helix-helix and strand-strand elements, under the assumption that the elements are contacting (Section 3.2.4). A probability of parallel or anti-parallel contact between two elements is provided by the coarse contact and orientation predictor (Section 3.2.3). One can thus evaluate two different probability measures of contact at the

Table 3.2: Average accuracy on long-range contacts for the element alignment predictor.

Class	E-E			H-H		
	L/5	L/10	Best 5	L/5	L/10	Best 5
All	0.24	0.25	0.25	0.09	0.10	0.10
All-Alpha	-	-	-	0.09	0.10	0.11
All-Beta	0.19	0.21	0.20	-	-	-
Alpha/Beta	0.22	0.19	0.21	0.07	0.07	0.07
Alpha+Beta	0.26	0.27	0.27	0.08	0.08	0.08

Class	E-E <sup>+</sup>			H-H <sup>+</sup>		
	L/5	L/10	Best 5	L/5	L/10	Best 5
All	0.35	0.36	0.37	0.11	0.12	0.13
All-Alpha	-	-	-	0.10	0.11	0.11
All-Beta	0.19	0.17	0.17	-	-	-
Alpha/Beta	0.52	0.55	0.54	0.11	0.14	0.12
Alpha+Beta	0.34	0.37	0.35	0.09	0.11	0.10

Contact prediction accuracy ( $Acc$ , see Section 3.2.1) of the element alignment predictor for long-range residue pairs. The length  $L$  refers to the sum of the lengths of helix/strand elements in the protein sequence. The protein domains having less than 5 contacts in the strand-strand and helix-helix regions have been excluded from the evaluation. We do not consider the strand-strand predictions on the All-Alpha class, as well as the helix-helix predictions on the All-Beta class. The performance on the strand-strand regions, E-E, E-E<sup>+</sup>, and helix-helix regions, H-H, H-H<sup>+</sup>, have been obtained by using the contact probabilities in Equations (3.10), (3.12), (3.11) and (3.13), respectively.

residue level for the alignment predictor: a naive measure that uses only the alignment scores, and a more refined measure that combines alignment and coarse scores. Specifically, consider two residues  $i$  and  $j$  in secondary structure elements  $S_n$  and  $S_m$ , where  $S_n$  and  $S_m$  are both helices or strands. A naive probability of contact between  $i$  and  $j$  can be derived from the alignment scores only by

$$p_{ij}^{\text{H-H}} = a_{PH} + a_{AH} \quad (3.10)$$

$$p_{ij}^{\text{E-E}} = a_{PE} + a_{AE} \quad (3.11)$$

where  $a_{PH}$  (helix-helix parallel contact),  $a_{AH}$  (helix-helix anti-parallel contact),  $a_{PE}$  (strand-strand parallel contact) and  $a_{AE}$  (strand-strand anti-parallel contact) are the contact proba-

bilities obtained with the alignment predictor for residues  $i$  and  $j$ . A more refined probability of contact can be defined by combining the alignment scores with the coarse predictor scores

$$p_{ij}^{\text{H-H}^+} = p_P \cdot a_{PH} + p_A \cdot a_{AH} \quad (3.12)$$

$$p_{ij}^{\text{E-E}^+} = p_P \cdot a_{PE} + p_A \cdot a_{AE} \quad (3.13)$$

where  $p_P$  and  $p_A$  are the probability of parallel and anti-parallel contact, obtained with the coarse contact predictor, between the secondary structure elements  $S_n$  and  $S_m$ .

The average accuracy on the 364 test domains for these two probability measures and for long-range residue pairs are reported in Table 3.2. The prediction accuracy is reported on the full set of protein domains (All), as well as on the main structural classes (All-Alpha, All-Beta, Alpha/Beta, Alpha+Beta). Overall, the prediction performance obtained by combining alignment and coarse probabilities (H-H<sup>+</sup> and E-E<sup>+</sup>) is higher than the one obtained by considering the alignment probabilities alone (H-H and E-E). Thus the coarse contact and alignment features alone contain relevant information on long-range residue-residue contacts, although the accuracy of this information is unevenly distributed with respect to the different structural classes and secondary structure elements. In particular, the prediction accuracy for beta-residues is much higher than for helix-residues, regardless of the structural class. This uneven distribution of performance is consistent with the native distribution of contacts between the respective classes of secondary structure elements: the strand-strand contacts are more dense than the helix-helix contacts and thus also easier to predict.

### 3.3.3 Residue-residue contact prediction: test set

We compare the performance on the 364 test domains of different contact predictors in order to separate the contribution of the deep-NN architecture from the contribution of the features

Table 3.3: Average accuracy and  $X_d$  comparison on long-range contacts.

Method	Acc			Xd		
	L/5	L/10	Best 5	L/5	L/10	Best 5
CMAPro	0.28	0.32	0.36	0.14	0.15	0.16
NN+CA	0.25	0.29	0.32	0.13	0.14	0.15
DNN	0.25	0.28	0.32	0.13	0.14	0.15
NN+C	0.23	0.27	0.30	0.12	0.13	0.14
NN+A	0.21	0.23	0.26	0.11	0.12	0.13
NN	0.20	0.24	0.26	0.10	0.12	0.13

Table 3.4: Average accuracy and  $X_d$  on long-range contacts for CMAPro.

Set	Acc			Xd		
	L/5	L/10	Best 5	L/5	L/10	Best 5
All	0.28	0.32	0.36	0.14	0.15	0.16
All-Alpha	0.20	0.22	0.25	0.12	0.13	0.13
All-Beta	0.28	0.31	0.36	0.12	0.13	0.15
Alpha/Beta	0.50	0.59	0.68	0.22	0.24	0.27
Alpha+Beta	0.27	0.32	0.36	0.14	0.15	0.16

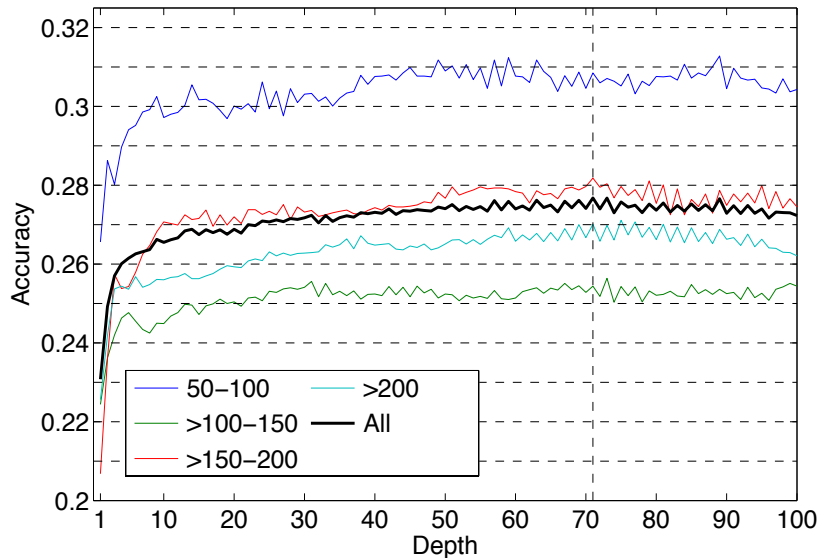


Figure 3.8: Accuracy (L/5 long range contacts) versus network depth for the set of test domains (All), the test domains of length between 50 and 100 residues (50-100), between 101 and 150 (>100-150), between 151 and 200 (> 150 – 200) and longer than 200 (>200). The dotted vertical line indicates the actual depth of CMAPro.

obtained with the coarse contact/orientation and alignment predictors (CA-features). Table 3.3 reports the performances of the full predictor (CMAPpro), a single-hidden-layer back-propagation neural network with CA-features (NN+CA) and without CA-features (NN), and a deep-NN architecture (DNN) that does not incorporate CA-features. In order to consider separately the contribution of coarse and alignment features, we also train a single-hidden-layer neural network that incorporates only coarse (NN+C) and only alignment (NN+A) features. For all such predictors, we build a corresponding ensemble by averaging the 10 cross-validation models. In Table 3.3, note that the performance of the basic neural network NN reflects the state-of-the-art in contact prediction, as assessed by all previous CASP experiments. Both the CA-features and the deep-NN architecture help improve the contact prediction accuracy in comparison to the performance of the plain neural network NN. The performance of the NN incorporating the CA-features (NN + CA) is indistinguishable from the performance of the deep-NN without CA-features (DNN). CMAPpro (deep-NN with CA-features) achieves the best performance among the predictors, indicating that both CA features and deep architecture play a role in improving contact prediction. Furthermore, in Table 3.3, the coarse features (NN+C) seem to be more informative than the alignment features (NN+A). On the other end, the performance comparison on the CASP datasets in the next Section shows that in specific cases the alignment features are more informative than the coarse features (compare NN+A versus NN+C in Table 3.5 and 3.7).

Table 3.4 shows the cross-validation performance of CMAPpro as a function of the main protein structural classes. This performances is somewhat consistent with what has been reported in literature: the residue contacts in the Alpha/Beta class are relatively easy to predict, while the contacts in the All-Alpha class are more difficult [27]. The 20% accuracy of CMAPpro on the All-Alpha class still represents some improvement with respect to the state-of-the-art for long-range contact prediction ( $\sim 15\%$ ) on this class of proteins [27].

The prediction performance of CMAPpro as a function of architecture depth is shown in



Table 3.5: Average  $Acc$  and  $X_d$  for seq. sep.  $\geq 24$  on CASP8 set.

Method	Acc			Xd		
	L/5	L/10	Best 5	L/5	L/10	Best 5
CMAPro	0.32	0.41	0.42	0.13	0.15	0.15
NN+CA	0.28	0.38	0.40	0.12	0.15	0.15
NN+A	0.26	0.33	0.32	0.11	0.12	0.14
DNN	0.25	0.35	0.37	0.11	0.13	0.14
RR157	0.24	0.30	0.32	0.09	0.10	0.11
RR072	0.24	0.30	0.28	0.11	0.13	0.13
NN+C	0.23	0.32	0.30	0.10	0.12	0.11
RR453	0.23	0.30	0.38	0.11	0.13	0.15
RR477	0.23	0.28	0.28	0.10	0.12	0.11
RR197	0.22	0.22	0.22	0.09	0.09	0.11
RR131	0.21	0.24	0.22	0.10	0.09	0.08
PSICOV	0.21	0.20	0.20	0.07	0.08	0.08
RR249	0.20	0.25	0.28	0.12	0.14	0.15
RR413	0.20	0.24	0.20	0.10	0.12	0.11
NN	0.20	0.25	0.27	0.09	0.10	0.10

Figure 3.8 for the full set of test domains (All), as well as for different subsets organized by domain lengths. Overall, the contact prediction accuracy improves up to depth  $\sim 50$  and then remains roughly constant for depths in the range of 50-100. Even for architectures with depth as large as 100, CMAPro does not show any sign of overfitting. The apparent weaker performance on domains of length  $>100-150$  is artificially due to an uneven distribution of the easiest targets across the different sets.

### 3.3.4 Residue-residue contact prediction: CASP test datasets

In addition to the top 8 CASP predictors, we include in the comparison also the recent mutual-information-based approach PSICOV, using multiple alignments obtained by running *jackhammer* (<http://hmmer.org>) for three iterations on the NR database [37]. The performance comparison on the CASP8 and CASP9 datasets are shown in Table 3.5 and Table 3.7, respectively.

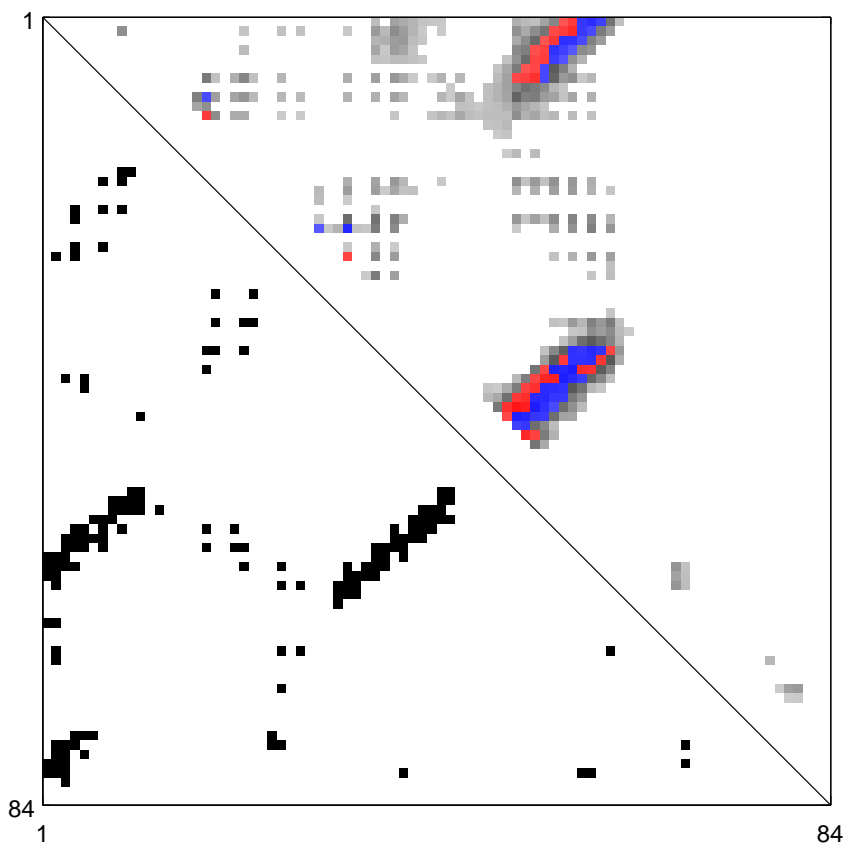


Figure 3.9: Native and predicted contact map for the T0604-D1 target from CASP9 set. The lower triangle shows the native contacts. The upper triangle shows contacts predicted by CMAPpro. The blue and red dots represent the correctly and incorrectly predicted contacts, respectively, among the L top-scored residue pairs.

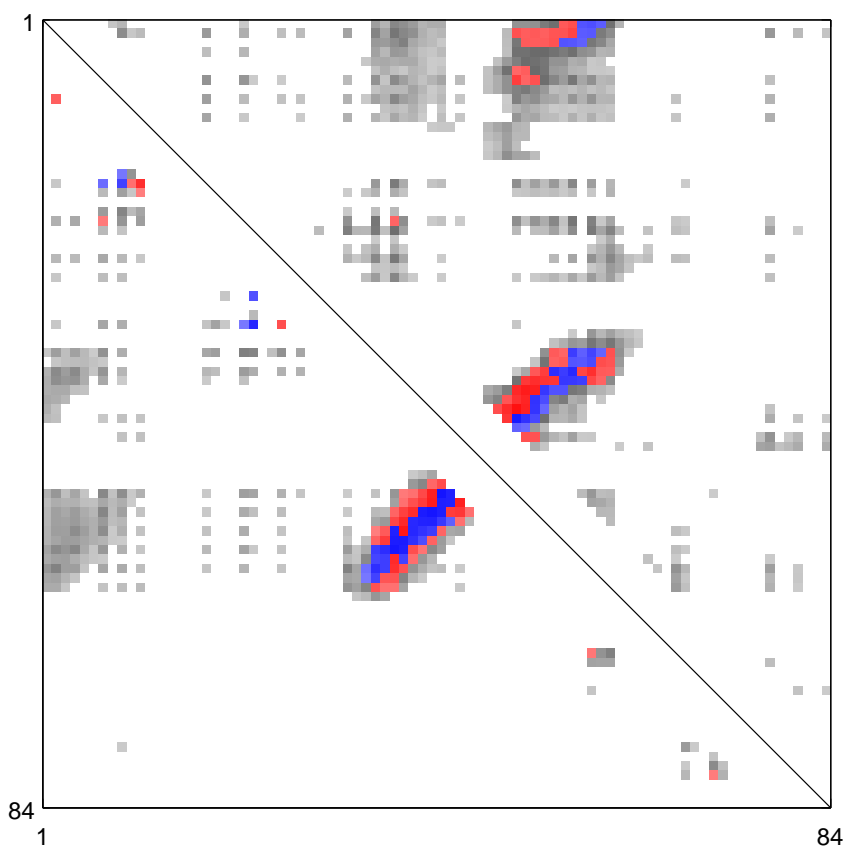


Figure 3.10: Predicted contact map for the T0604-D1 target from CASP9 dataset. The lower triangle shows the predictions obtained with DNN and the upper triangle those obtained with NN+CA. The blue and red dots represent the correctly and incorrectly predicted contacts, respectively, among the  $L$  top-scored residue pairs.

Table 3.6: Average  $Acc$  and  $X_d$  for seq. sep.  $\geq 24$  on CASP9 set.

Method	Acc			Xd		
	L/5	L/10	Best 5	L/5	L/10	Best 5
RR490	0.32	0.37	0.44	0.15	0.17	0.20
CMAPro	0.31	0.35	0.34	0.13	0.15	0.15
DNN	0.27	0.31	0.41	0.12	0.14	0.16
NN+CA	0.23	0.27	0.29	0.11	0.12	0.13
RR051	0.22	0.24	0.24	0.11	0.12	0.12
RR103	0.21	0.27	0.31	0.10	0.12	0.12
NN+C	0.21	0.27	0.25	0.10	0.11	0.11
RR002	0.21	0.23	0.23	0.11	0.12	0.12
PSICOV	0.20	0.28	0.33	0.08	0.10	0.11
NN+A	0.20	0.20	0.21	0.09	0.09	0.09
RR138	0.19	0.23	0.26	0.09	0.11	0.11
NN	0.19	0.19	0.19	0.09	0.09	0.09
RR375	0.18	0.21	0.24	0.08	0.09	0.10
RR204	0.18	0.20	0.22	0.09	0.10	0.11
RR422	0.17	0.20	0.21	0.09	0.10	0.09

On the CASP datasets the performance improvements obtained by considering separately the coarse/orientation and alignment features (NN+CA) and the deep-NN architecture (DNN) are somewhat different from those in Table 3.3. NN+CA performs better on the CASP8 dataset, whereas DNN performs better on the CASP9 dataset. CMAPro combines and refines the qualities of these two predictors achieving higher accuracy on both the CASP8 and CASP9 datasets. This behavior can be explained by considering an example. Figures 3.9 and 3.10 show the predicted contacts for the CASP9 domain T0604-D1. The red and blue dots in the picture represent the  $L$  top-scored true positive and false positive contacts, respectively. The predictions obtained by DNN and NN + CA are compared in Figure 3.10. Globally, the two predictors assign a high probability of contact (grey dots) to approximately the same regions. Locally, however, they assign different contact probabilities to the individual pairs of residues, leading to different sets of correctly predicted contacts (blue dots). CMAPro combines and refines the characteristics of these two predictors (Fig. 3.9): the segmentsegment features improve the identification of contacting regions between secondary

structure elements and the DNN is able to refine the prediction scores. Compared to other methods, CMAPpro is considerably more accurate on both CASP datasets. In particular, on the CASP8 dataset, CMAPpro achieves the best ranking both in terms of  $Acc$  and  $X_d$ . The only method [86] outperforming CMAPpro on the CASP9 dataset by a small margin relies on 3D structure models for deriving contact predictions through consensus, which defeats the purpose of predicting contact maps from scratch. Indeed, if we remove the only 3 TBM domains from the CASP9 dataset and focus exclusively on the FM targets which are harder to predict, then RR490's accuracy ( $L/5$ ) drops down from 0.32 to 0.28 while CMAPpro's accuracy increases from 0.31 to 0.32.

Due to the small number of targets, the average performances on the CASP8 and CASP9 are consistently affected by the network depth (Figure 3.11). In particular, on the CASP8 set, for architectures depths in the range of 10-100, the average accuracy on  $L/5$  long range contacts varies from 0.30 to 0.35. On the CASP9 set, the average accuracy varies from 0.28 to 0.31. Notwithstanding such variability, on both CASP datasets, the performance of CMAPpro remains above the performance of the other methods at all depth values. As a general trend, on both CASP8 and CASP9 datasets the improvement obtained in contact prediction with CMAPpro is  $\sim 10\%$  or higher with respect to methods that do not use 3D structures. In Table 3.5 and 3.7, we also note that the performance of the plain neural network predictor NN is comparable to the average performance across all groups. This confirms that the overall good performance of CMAPpro is not due to the particular set of protein domains used for training.

The accuracy of PSICOV ( $\sim 20\%$ ) is lower than previously reported ( $>50\%$ ) [37]. The performance of PSICOV is considerably affected by the quality of the multiple alignments. Since TBM/FM targets for contact prediction at CASP usually have few homologs in the protein sequence databases, this considerably lowers the prediction accuracy of PSICOV. The performance of PSICOV may suggest that even the most updated database of protein

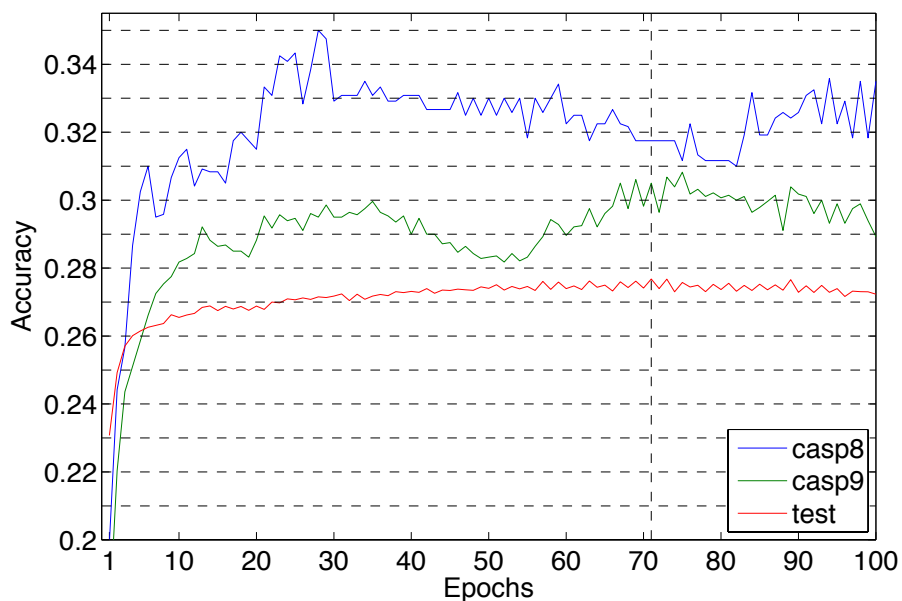


Figure 3.11: Accuracy (L/5 long range contacts) versus network depth for the set of test domains (test), CASP8 domains (casp8) and CASP9 domains (casp9). The dotted vertical line indicates the actual depth of CMAPpro.

sequences (i.e. the NR database used to extract sequence profiles) does not contain enough information to derive rich evolutionary profiles for the CASP hardest targets. On the other hand, PSICOV relies only on multiple alignments and thus a direct comparison with methods that make use of predicted secondary structure or solvent accessibility is somewhat unfair.

Finally, Tables 3.7 and 3.8 report the head-to-head comparison of the ten top predictors on the CASP data. These results show that the average accuracies of the best performing methods are not biased by just a few very good predictions. With very few exceptions, in head-to-head comparisons, CMAPpro achieves a better accuracy for over 60% of the targets and worse accuracy for less than 30% of the targets.

Table 3.7: CASP8 1-to-1 comparison of top ten predictors

	CMAPpro	RR157	RR072	RR453	RR477	RR197	RR131	PSICOV	RR249	RR413
CMAPpro	-	0.58	0.42	0.83	0.67	0.67	0.75	0.75	0.67	0.67
RR157	0.33	-	0.42	0.33	0.50	0.58	0.50	0.67	0.42	0.58
RR072	0.33	0.33	-	0.50	0.42	0.42	0.58	0.83	0.50	0.58
RR453	0.17	0.58	0.42	-	0.50	0.58	0.50	0.75	0.50	0.67
RR477	0.17	0.42	0.42	0.33	-	0.58	0.42	0.50	0.50	0.50
RR197	0.25	0.33	0.42	0.42	0.25	-	0.50	0.50	0.33	0.42
RR131	0.25	0.33	0.42	0.25	0.42	0.33	-	0.58	0.33	0.33
PSICOV	0.25	0.33	0.17	0.25	0.42	0.50	0.42	-	0.42	0.33
RR249	0.08	0.42	0.33	0.25	0.33	0.50	0.50	0.58	-	0.58
RR413	0.08	0.42	0.42	0.33	0.42	0.33	0.42	0.58	0.33	-

The entries of the table show the percentage of targets for which the *Acc* of the method in the row is strictly higher than the *Acc* of the method in the column. The *Acc* is computed for L/5 pairs and sequence separation  $\geq 24$ .

Table 3.8: CASP9 1-to-1 comparison of top ten predictors

	RR490	CMAPpro	RR051	RR103	RR002	PSICOV	RR138	RR375	RR204	RR422
RR490	-	0.50	0.64	0.61	0.61	0.82	0.64	0.64	0.68	0.68
CMAPpro	0.46	-	0.61	0.71	0.61	0.86	0.61	0.71	0.64	0.71
RR051	0.29	0.21	-	0.50	0.43	0.82	0.46	0.57	0.57	0.61
RR103	0.29	0.11	0.36	-	0.36	0.64	0.43	0.39	0.46	0.43
RR002	0.32	0.25	0.36	0.46	-	0.68	0.46	0.39	0.54	0.50
PSICOV	0.14	0.11	0.07	0.25	0.25	-	0.21	0.21	0.21	0.18
RR138	0.25	0.18	0.32	0.46	0.32	0.68	-	0.57	0.54	0.50
RR375	0.25	0.14	0.32	0.29	0.46	0.75	0.32	-	0.32	0.46
RR204	0.18	0.25	0.32	0.36	0.29	0.71	0.36	0.39	-	0.36
RR422	0.25	0.14	0.21	0.29	0.32	0.71	0.32	0.36	0.39	-

The entries of the table show the percentage of targets for which the *Acc* of the method in the row is strictly higher than the *Acc* of the method in the column. The *Acc* is computed for L/5 pairs and sequence separation  $\geq 24$ .

## 3.4 Conclusion

Here we have introduced a new approach for the prediction of protein contact maps. In particular, partly inspired by the observation that nature uses an iterative refinement approach to “compute” the structure of proteins, we have developed modular deep architectures that can integrate information over multiple temporal and spatial scales. In rigorous tests, these architectures have been shown to predict contact maps with an accuracy close to 30%, a significant improvement. Although further improvements are necessary, it should be obvious that there are many generalizations and variations on the architectures and training methods we have described that remain to be explored, giving us hope that further progress lies ahead.

# Chapter 4

## Conclusion

We described new machine learning approaches for the side-chain prediction and the contact prediction. These predictors had a better performance than the existing methods.

We described the new predictor, SIDEpro. Accuracy was improved by using neural networks as energy functions between two atoms in contact. Fast identification of atom contacts using boxes in 3D space also aided in improving the speed of SIDEpro. SIDEpro had a slightly better accuracy than the latest SCWRL, and SIDEpro was 7 to 20 times faster in its predictions. In addition, we created the rotamer library for FPTMs which enabled SIDEpro to predict the side-chains of FPTMs. SIDEpro can also predict any other non-standard amino acids by creating rotamers with COSMOS. The accuracy for FPTMs and non-standard amino acids is roughly comparable to its accuracy on the natural amino acids.

We could improve the contact predictor, CMAPpro, by using new deep neural networks architecture and the results of two predictors, coarse contact and orientation prediction and element alignment prediction. The accuracy of this method was close to 30% for long-range contacts on CASP8 and CASP9 datasets. This is a relevant improvement over the state-of-the-art in contact prediction.



Since both SIDEpro and CMAPpro show better performances compared to the other existing methods, the machine learning approach with neural networks worked well for the protein structure prediction. Although the protein structure prediction is a difficult and unsolved problem, machine learning has undoubtedly improved its accuracy. They are part of the SCRATCH suit of predictors and are available from : <http://scratch.proteomics.ics.uci.edu/>.

# Bibliography

- [1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J. Mol. Biol.*, 215(3):403–410, Oct 1990.
- [2] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, 25(17):3389–3402, Sep 1997.
- [3] A. Andronico, A. Randall, R. Benz, and P. Baldi. Data-driven high-throughput prediction of the 3-d structure of small molecules: review and progress. *Journal of Chemical Information and Modeling*, 51(4):760–766, apr 2011.
- [4] N. Andrusier, E. Mashiach, R. Nussinov, and H. J. Wolfson. Principles of flexible protein-protein docking. *Proteins: Structure, Function, and Bioinformatics*, 73(2):271–289, 2008.
- [5] A. Bairoch and R. Apweiler. The swiss-prot protein sequence database and its supplement trembl in 2000. *Nucleic Acids Research*, 28(1):45–48, 2000.
- [6] P. Baldi and S. Brunak. *Bioinformatics: the machine learning approach*. MIT Press, 2001.
- [7] E. T. Baldwin, I. T. Weber, R. St Charles, J. C. Xuan, E. Appella, M. Yamada, K. Matsushima, B. F. Edwards, G. M. Clore, and A. M. Gronenborn. Crystal structure of interleukin 8: symbiosis of NMR and crystallography. *Proc. Natl. Acad. Sci. U.S.A.*, 88(2):502–506, Jan 1991.
- [8] H. Berman, T. Battistuz, T. Bhat, W. Bluhm, P. Bourne, K. Burkhardt, Z. Feng, G. Gilliland, L. Iype, S. Jain, P. Fagan, J. Marvin, D. Padilla, V. Ravichandran, B. Schneider, N. Thanki, H. Weissig, J. Westbrook, and C. Zardecki. The protein data bank. *Acta Crystallographica, Section D: Biological Crystallography*, 58:899–907, 2002.
- [9] M. S. Bhuyan and X. Gao. A protein-dependent side-chain rotamer library. *BMC Bioinformatics*, 12 Suppl 14:S10, 2011.
- [10] P. Bjorkholm, P. Daniluk, A. Kryshchovych, K. Fidelis, R. Andersson, and T. R. Hvidsten. Using multi-data hidden Markov models trained on local neighborhoods of protein

- structure to predict residue-residue contacts. *Bioinformatics*, 25(10):1264–1270, May 2009.
- [11] N. Blom, S. Gammeltoft, and S. Brunak. Sequence and structure-based prediction of eukaryotic protein phosphorylation sites. *Journal of Molecular Biology*, 294(5):1351 – 1362, 1999.
- [12] N. Blom, T. Sicheritz-Pontén, R. Gupta, S. Gammeltoft, and S. Brunak. Prediction of post-translational glycosylation and phosphorylation of proteins from the amino acid sequence. *Proteomics*, 4(6):1633–1649, 2004.
- [13] M. J. Bower, F. E. Cohen, and R. L. Dunbrack, Jr. Prediction of protein side-chain rotamers from a backbone-dependent rotamer library: A new homology modeling tool. *J Mol Biol*, 267:1268–1282, 1997.
- [14] L. Burger and E. van Nimwegen. Disentangling direct from indirect co-evolution of residues in protein alignments. *PLoS Comput. Biol.*, 6(1):e1000633, Jan 2010.
- [15] A. A. Canutescu, A. A. Shelenkov, and R. L. Dunbrack, Jr. A graph-theory algorithm for rapid protein side-chain prediction. *Protein Sci*, 12(9):2001–2004, sep 2003.
- [16] J. M. Chandonia, G. Hon, N. S. Walker, L. Lo Conte, P. Koehl, M. Levitt, and S. E. Brenner. The ASTRAL Compendium in 2004. *Nucleic Acids Res.*, 32(Database issue):D189–192, Jan 2004.
- [17] J. Cheng and P. Baldi. Improved residue contact prediction using support vector machines and a large feature set. *BMC Bioinformatics*, 8:113, 2007.
- [18] J. Cheng, A. Randall, M. Sweredoski, and P. Baldi. Scratch: a protein structure and structural feature prediction server. *Nucleic Acids Res*, 33:W72–76, 2005.
- [19] H. Dinkel, C. Chica, A. Via, C. M. Gould, L. J. Jensen, T. J. Gibson, and F. Diella. Phospho.elm: a database of phosphorylation sites update 2011. *Nucleic Acids Research*, 2010.
- [20] R. L. Dunbrack, Jr. Rotamer libraries in the 21st century. *Curr Opin Struct Biol*, 12:431–440, 2002.
- [21] R. L. Dunbrack, Jr. and F. E. Cohen. Bayesian statistical analysis of protein sidechain rotamer preferences. *Protein Sci*, 6:1661–1681, 1997.
- [22] R. L. Dunbrack, Jr. and M. Karplus. Backbone-dependent rotamer library for proteins: Application to side-chain prediction. *J Mol Biol*, 230:543–574, 1993.
- [23] R. L. Dunbrack, Jr. and M. Karplus. Conformational analysis of the backbone-dependent rotamer preferences of protein sidechains. *Nat Struct Biol*, 1:334–340, 1994.
- [24] S. D. Dunn, L. M. Wahl, and G. B. Gloor. Mutual information without the influence of phylogeny or entropy dramatically improves residue contact prediction. *Bioinformatics*, 24(3):333–340, Feb 2008.

- [25] E. Eyal, R. Najmanovich, B. J. Mcconkey, M. Edelman, and V. Sobolev. Importance of solvent accessibility and contact surfaces in modeling side-chain conformations in proteins. *Journal of Computational Chemistry*, 25(5):712–724, 2004.
- [26] I. Ezkurdia, O. Grana, J. M. Izarzugaza, and M. L. Tress. Assessment of domain boundary predictions and the prediction of intramolecular contacts in CASP8. *Proteins*, 77 Suppl 9:196–209, 2009.
- [27] P. Fariselli, A. Eusebi, P. L. Martelli, D. T. Jones, and R. Casadio. Improving the prediction of helix-residue contacts in all-alpha proteins. In *Proceedings of the 9th WSEAS International Conference on Neural Networks*, NN’08, pages 89–94, Stevens Point, Wisconsin, USA, 2008. World Scientific and Engineering Academy and Society (WSEAS).
- [28] P. Fariselli, O. Olmea, A. Valencia, and R. Casadio. Progress in predicting inter-residue contacts of proteins with neural networks and correlated mutations. *Proteins*, Suppl 5:157–162, 2001.
- [29] D. Gfeller, O. Michielin, and V. Zoete. Expanding molecular modeling and design tools to non-natural sidechains. *J Comput Chem*, 33(18):1525–1535, Jul 2012.
- [30] F. Gnad, J. Gunawardena, and M. Mann. Phosida 2011: the posttranslational modification database. *Nucleic Acids Research*, 39(suppl 1):D253–D260, 2011.
- [31] U. Gobel, C. Sander, R. Schneider, and A. Valencia. Correlated mutations and residue contacts in proteins. *Proteins*, 18(4):309–317, Apr 1994.
- [32] R. Goldstein. Efficient rotamer elimination applied to protein side-chains and related spin glasses. *Biophys J*, 66:1335–1340, 1994.
- [33] J. J. Gray, S. Moughon, C. Wang, O. Schueler-Furman, B. Kuhlman, C. A. Rohl, and D. Baker. Protein-protein docking with simultaneous optimization of rigid-body displacement and side-chain conformations. *Journal of Molecular Biology*, 331(1):281 – 299, 2003.
- [34] S. Hamby and J. Hirst. Prediction of glycosylation sites using random forests. *BMC Bioinformatics*, 9(1):500, 2008.
- [35] C. Hartmann, I. Antes, and T. Lengauer. Ireces: a new algorithm for the selection of most probable ensembles of side-chain conformations in protein models. *Protein Science*, 16(7):1294–1307, jul 2007.
- [36] P. V. Hornbeck, J. M. Kornhauser, S. Tkachev, B. Zhang, E. Skrzypek, B. Murray, V. Latham, and M. Sullivan. Phosphositeplus: a comprehensive resource for investigating the structure and function of experimentally determined post-translational modifications in man and mouse. *Nucleic Acids Research*, 40(D1):D261–D270, 2012.

- [37] D. T. Jones, D. W. Buchan, D. Cozzetto, and M. Pontil. PSICOV: precise structural contact prediction using sparse inverse covariance estimation on large multiple sequence alignments. *Bioinformatics*, 28(2):184–190, Jan 2012.
- [38] K. Julenius, A. Mølgaard, R. Gupta, and S. Brunak. Prediction, conservation analysis, and structural characterization of mammalian mucin-type O-glycosylation sites. *Glycobiology*, 15(2):153–164, Feb 2005.
- [39] J. C. KENDREW, G. BODO, H. M. DINTZIS, R. G. PARRISH, H. WYCKOFF, and D. C. PHILLIPS. A three-dimensional model of the myoglobin molecule obtained by x-ray analysis. *Nature*, 181(4610):662–666, Mar 1958.
- [40] T. S. Keshava Prasad, R. Goel, K. Kandasamy, S. Keerthikumar, S. Kumar, S. Mathivanan, D. Telikicherla, R. Raju, B. Shafreen, A. Venugopal, L. Balakrishnan, A. Marimuthu, S. Banerjee, D. S. Somanathan, A. Sebastian, S. Rani, S. Ray, C. J. Harrys Kishore, S. Kanth, M. Ahmed, M. K. Kashyap, R. Mohmood, Y. L. Ramachandra, V. Krishna, B. A. Rahiman, S. Mohan, P. Ranganathan, S. Ramabadran, R. Chaerkady, and A. Pandey. Human protein reference database2009 update. *Nucleic Acids Research*, 37(suppl 1):D767–D772, 2009.
- [41] G. A. Khoury, R. C. Baliban, and C. A. Floudas. Proteome-wide post-translational modification statistics: frequency analysis and curation of the swiss-prot database. *Scientific Reports*, 1(90), 2011.
- [42] J. H. Kim, J. Lee, B. Oh, K. Kimm, and I. Koh. Prediction of phosphorylation sites using svms. *Bioinformatics*, 20(17):3179–3184, 2004.
- [43] C. Kingsford, B. Chazelle, and M. Singh. Solving and analyzing side-chain positioning problems using linear and integer programming. *Bioinformatics*, 21:1028–1036, 2005.
- [44] P. Koehl and M. Delarue. Application of a self-consistent mean field theory to predict protein side-chains conformation and estimate their conformational entropy. *J Mol Biol*, 239:249–275, 1994.
- [45] G. G. Krivov, M. V. Shapovalov, and R. L. Dunbrack, Jr. Improved prediction of protein side-chain conformations with scwrl4. *Proteins*, 77(4):778–95, dec 2009.
- [46] A. Kryshtafovych, K. Fidelis, and J. Moult. CASP9 results compared to those of previous CASP experiments. *Proteins*, 79 Suppl 10:196–207, 2011.
- [47] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin. Exploring strategies for training deep neural networks. *J. Mach. Learn. Res.*, 10:1–40, June 2009.
- [48] A. Leaver-Fay, M. Tyka, S. M. Lewis, O. F. Lange, J. Thompson, R. Jacak, K. Kaufman, P. D. Renfrew, C. A. Smith, W. Sheffler, I. W. Davis, S. Cooper, A. Treuille, D. J. Mandell, F. Richter, Y. E. Ban, S. J. Fleishman, J. E. Corn, D. E. Kim, S. Lyskov, M. Berrondo, S. Mentzer, Z. Popović, J. J. Havranek, J. Karanicolas, R. Das, J. Meiler, T. Kortemme, J. J. Gray, B. Kuhlman, D. Baker, and P. Bradley. ROSETTA3: an

- object-oriented software suite for the simulation and design of macromolecules. *Meth. Enzymol.*, 487:545–574, 2011.
- [49] S. Li, H. Li, M. Li, Y. Shyr, L. Xie, and Y. Li. Improved prediction of lysine acetylation by support vector machines. *Protein Pept. Lett.*, 16(8):977–983, 2009.
- [50] S. Li, B. Liu, R. Zeng, Y. Cai, and Y. Li. Predicting O-glycosylation sites in mammalian proteins by using SVMs. *Comput Biol Chem*, 30(3):203–208, Jun 2006.
- [51] S. Liang and N. Grishin. Side-chain modeling with an optimized scoring function. *Protein Sci*, 11:322–331, 2002.
- [52] S. Liang, D. Zheng, C. Zhang, and D. Standley. Fast and accurate prediction of protein side-chain conformations. *Bioinformatics*, 27(20):2913–2914, oct 2011.
- [53] S. C. Lovell, J. M. Word, J. S. Richardson, and D. C. Richardson. The penultimate rotamer library. *Proteins*, 40(3):389–408, Aug 2000.
- [54] M. Lu, A. D. Dousis, and J. Ma. Opus-rota: A fast and accurate method for side-chain modeling. *Protein Science*, 17(9):1576–1585, 2008.
- [55] D. MacKay. Bayesian non-linear modelling for the prediction competition. In *In ASHRAE Transactions, V.100, Pt.2*, pages 1053–1062. ASHRAE, 1994.
- [56] J. Mendes, C. M. Soares, and M. A. Carrondo. Improvement of side-chain modeling in proteins with the self-consistent mean field theory method based on an analysis of the factors influencing prediction. *Biopolymers*, 50(2):111–131, 1999.
- [57] K. M. Misura, D. Chivian, C. A. Rohl, D. E. Kim, and D. Baker. Physically realistic homology models built with ROSETTA can be more accurate than their templates. *Proc. Natl. Acad. Sci. U.S.A.*, 103(14):5361–5366, Apr 2006.
- [58] B. Monastyrskyy, K. Fidelis, A. Tramontano, and A. Kryshtafovych. Evaluation of residue-residue contact predictions in CASP9. *Proteins*, 79 Suppl 10:119–125, 2011.
- [59] D. Muramatsu, M. Kondo, M. Sasaki, S. Tachibana, and T. Matsumoto. A markov chain monte carlo algorithm for bayesian dynamic signature verification. *IEEE T Inf Foren Sec*, 1:22–34, 2006.
- [60] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, 247(4):536–540, Apr 1995.
- [61] K. Nagata, A. Randall, and P. Baldi. SIDEpro: a novel machine learning approach for the fast and accurate prediction of side-chain conformations. *Proteins*, 80(1):142–153, Jan 2012.
- [62] R. M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag, 1996.

- [63] J. C. Obenauer, L. C. Cantley, and M. B. Yaffe. Scansite 2.0: proteome-wide prediction of cell signaling interactions using short sequence motifs. *Nucleic Acids Research*, 31(13):3635–3641, 2003.
- [64] N. O’Boyle, M. Banck, C. James, C. Morley, T. Vandermeersch, and G. Hutchison. Open babel: An open chemical toolbox. *Journal of Chemoinformatics*, 3:33, 2011.
- [65] O. Olmea and A. Valencia. Improving contact predictions by the combination of correlated mutations and other sources of sequence information. *Fold Des*, 2(3):25–32, 1997.
- [66] D. Plewczynski, S. Basu, and I. Saha. AMS 4.0: consensus prediction of post-translational modifications in protein sequences. *Amino Acids*, May 2012.
- [67] G. Pollastri, P. Baldi, P. Fariselli, and R. Casadio. Prediction of coordination number and relative solvent accessibility in proteins. *Proteins*, 47(2):142–153, May 2002.
- [68] G. Pollastri, D. Przybylski, B. Rost, and P. Baldi. Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. *Proteins*, 47(2):228–235, May 2002.
- [69] G. Pollastri, A. Vullo, P. Frasconi, and P. Baldi. Modular DAG-RNN architectures for assembling coarse protein structures. *J. Comput. Biol.*, 13(3):631–650, Apr 2006.
- [70] M. Punta and B. Rost. PROFcon: novel prediction of long-range contacts. *Bioinformatics*, 21(13):2960–2968, Jul 2005.
- [71] J. Ren, X. Gao, C. Jin, M. Zhu, X. Wang, A. Shaw, L. Wen, X. Yao, and Y. Xue. Systematic study of protein sumoylation: Development of a site-specific predictor of sumosp 2.0. *Proteomics*, 9(12):3409–3412, 2009.
- [72] P. D. Renfrew, E. J. Choi, R. Bonneau, and B. Kuhlman. Incorporation of noncanonical amino acids into Rosetta and use in computational protein-peptide interface design. *PLoS ONE*, 7(3):e32637, 2012.
- [73] C. A. Rohl, C. E. M. Strauss, D. Chivian, and D. Baker. Modeling structurally variable regions in homologous proteins with rosetta. *Proteins*, 55:656–677, 2004.
- [74] A. Sali and T. Blundell. Comparative protein modeling by satisfaction of spatial restraints. *Journal of Molecular Biology*, 234(3):779–815, 1993.
- [75] A. D. Scouras and V. Daggett. The DYNAMEOmics rotamer library: amino acid side chain conformations and dynamics from comprehensive molecular dynamics simulations in water. *Protein Sci.*, 20(2):341–352, Feb 2011.
- [76] G. Shackelford and K. Karplus. Contact prediction using mutual information and neural nets. *Proteins*, 69 Suppl 8:159–164, 2007.

- [77] M. V. Shapovalov and R. L. Dunbrack, Jr. A smoothed backbone-dependent rotamer library for proteins derived from adaptive kernel density estimates and regressions. *Structure*, 19(6):844–858, jun 2011.
- [78] K. Simons, I. Ruczinski, C. Kooperberg, B. Fox, C. Bystroff, and D. Baker. Improved recognition of native-like protein structures using a combination of sequence-dependent and sequence-independent features of proteins. *Proteins*, 34:82–95, 1999.
- [79] J. Skolnick, D. Kihara, and Y. Zhang. Development and large scale benchmark testing of the PROSPECTOR\_3 threading algorithm. *Proteins*, 56(3):502–518, Aug 2004.
- [80] W. R. Taylor, D. T. Jones, and M. I. Sadowski. Protein topology from predicted residue contacts. *Protein Sci.*, 21(2):299–305, Feb 2012.
- [81] M. L. Tress and A. Valencia. Predicted residue-residue contacts can help the scoring of 3D models. *Proteins*, 78(8):1980–1991, Jun 2010.
- [82] A. Vullo and P. Frasconi. Prediction of protein coarse contact maps. *J Bioinform Comput Biol*, 1(2):411–431, Jul 2003.
- [83] A. Vullo, I. Walsh, and G. Pollastri. A two-stage approach for improved prediction of residue contact maps. *BMC Bioinformatics*, 7:180, 2006.
- [84] J. Wan, S. Kang, C. Tang, J. Yan, Y. Ren, J. Liu, X. Gao, A. Banerjee, L. B. Ellis, and T. Li. Meta-prediction of phosphorylation sites with weighted voting and restricted grid search parameter selection. *Nucleic Acids Res.*, 36(4):e22, Mar 2008.
- [85] Q. Wang, A. R. Parrish, and L. Wang. Expanding the genetic code for biological studies. *Chem. Biol.*, 16(3):323–336, Mar 2009.
- [86] Z. Wang, J. Eickholt, and J. Cheng. MULTICOM: a multi-level combination approach to protein structure prediction and its assessments in CASP8. *Bioinformatics*, 26(7):882–888, Apr 2010.
- [87] M. P. Williamson, T. F. Havel, and K. Wuthrich. Solution conformation of proteinase inhibitor IIA from bull seminal plasma by <sup>1</sup>H nuclear magnetic resonance and distance geometry. *J. Mol. Biol.*, 182(2):295–315, Mar 1985.
- [88] Z. Xiang and B. Honig. Extending the accuracy limits of prediction for side-chain conformations. *J Mol Biol*, 311:421–430, 2001.
- [89] Z. Xiang, P. J. Steinbach, M. P. Jacobson, R. A. Friesner, and B. Honig. Prediction of side-chain conformations on protein surfaces. *Proteins: Structure, Function, and Bioinformatics*, 66(4):814–823, 2007.
- [90] J. Xie and P. G. Schultz. Adding amino acids to the genetic repertoire. *Curr Opin Chem Biol*, 9(6):548–554, Dec 2005.



- [91] J. Xu. Rapid protein side-chain packing via tree decomposition. In S. Miyano, J. Mesirov, S. Kasif, S. Istrail, P. Pevzner, and M. Waterman, editors, *Research in Computational Molecular Biology*, volume 3500 of *Lecture Notes in Computer Science*, pages 423–439. Springer Berlin / Heidelberg, 2005.
- [92] J. Xu, Y. He, B. Qiang, J. Yuan, X. Peng, and X.-M. Pan. A novel method for high accuracy sumoylation site prediction from protein sequences. *BMC Bioinformatics*, 9(1):8, 2008.
- [93] Y. Zhang, A. Kolinski, and J. Skolnick. Touchstone:ii a new approach to ab initio protein structure prediction. *Biophys J*, 85:1145–1164, 2003.
- [94] M. Zhichao, Y. Cao, and T. Jiang. Rasp:rapid modeling of protein side-chain conformations. *Bioinformatics*, 27(22):3117–3122, sep 2011.