BAYESIAN MODELS AND ALGORITHMS FOR PROTEIN SECONDARY STRUCTURE AND BETA-SHEET PREDICTION

A Thesis
Presented to
The Academic Faculty

by

Zafer Aydın

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
December 2008

BAYESIAN MODELS AND ALGORITHMS FOR PROTEIN SECONDARY STRUCTURE AND BETA-SHEET PREDICTION

Approved by:

Dr. Yucel Altunbasak, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Ghassan Alregib
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Mark Borodovsky
The Wallace H. Coulter Department of Biomedical Engineering, Computational Science and Engineering Division of the College of Computing
Georgia Institute of Technology

Dr. James McClellan
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Russel Mersereau
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Brani Vidakovic
The Wallace H. Coulter Department of Biomedical Engineering
Georgia Institute of Technology

Date Approved: 3 September 2008

To my dear wife Ayten

ACKNOWLEDGEMENTS

I am grateful to my advisor, Dr. Yücel Altunbaşak for giving me the courage to work on bioinformatics. I feel so lucky to have such an intelligent advisor. His advice shed light on my cloudy days and guided me in all aspects of my life. He is also a very reasonable and a considerate person. He has been a friend more than a boss and supported me in all aspects of my PhD. I would like to thank my co-advisor, Dr. Mark Borodovsky. He helped me to define my research problem and discussed many aspects of my research. His guidance and support helped me to adapt to the field easily. He also introduced me to his group, which had a very warm and friendly atmosphere. I want to thank Dr. Russell M. Mersereau, Dr. Ghassan Al-Regib, Dr. James McClellan, and Dr. Brani Vidakovic for serving in my committee. Special thanks go to Dr. Hakan Erdogan in Sabanci University, Turkey for his collaboration and invaluable feedback. He is such a nice person.

During my PhD, I enjoyed the friendship of so many great people. Among those, I would like to express my gratitude to Ali Cafer Gürbüz and Sevgi Zübeyde Gürbüz for their friendship. They are truly golden-hearted people. I am also grateful to all my colleagues and friends at Georgia Tech. I would like to thank my friends in the Multimedia Computing and Communications Lab, Dr. Borodovsky's Lab, as well as the members of the Center for Signal and Image Processing. Especially, I would like to thank Dr. Alex Lomsadze in Dr. Borodovsky's lab for sharing his experience. He is a real scientist.

I am thankful to the CSIP staff for providing such a pleasant work environment. I would like to particularly thank Keith May, Fanchette Hillery, Christy Ellis, and Lisa Gardner.

Finally, I would like to thank my mother, my father, and my sister for being there all the time. They were supportive, caring, and encouraging throughout this journey. I cannot thank enough to my dear wife Ayten Aydın for her love, patience and support. She is my second half.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

In this thesis, we developed Bayesian models and machine learning algorithms for protein secondary structure and β-sheet prediction problems. In protein secondary structure prediction, we developed hidden semi-Markov models, N-best algorithms and training set reduction procedures for proteins in the single-sequence category. We introduced three residue dependency models (both probabilistic and heuristic) incorporating the statistically significant amino acid correlation patterns at structural segment borders. In those models, we allowed dependencies to positions outside the segments to relax the condition of segment independence. Another novelty of the models is the dependency to downstream positions, which is important due to asymmetric correlation patterns observed uniformly in structural segments. Apart from the more elaborate dependency structure, we introduced a training set reduction strategy to refine estimates of the model parameters. Among the dataset reduction methods, the composition based reduction technique with thresholding generated the most accurate results in the single-sequence setting. To incorporate non-local interactions characteristic of β-sheets into the secondary structure prediction method, we developed two N-best algorithms and a Bayesian β-sheet model. We showed that the information in suboptimal segmentations is useful and can improve the sensitivity of the Viterbi algorithm. We also investigated the effect of incorporating non-local interactions into the single-sequence prediction method.

In β-sheet prediction, we developed a Bayesian model to characterize the conformational organization of β-sheets and efficient algorithms to compute the optimum architecture, which includes β-strand pairings, interaction types (parallel or anti-parallel) and residue-residue interactions (contact maps). We analyzed proteins

according to the number of β-strands they contain. We introduced a Bayesian approach for proteins with six or less β-strands, in which we modeled the conformational features in a probabilistic framework by combining the amino acid pairing potentials with *a priori* knowledge of β-strand organizations. To select the optimum β-sheet architecture, we analyzed the space of possible conformations by efficient heuristics, in which we significantly reduce the search space by enforcing the amino acid pairs that have strong interaction potentials. Furthermore, we employed an algorithm that finds the optimum pairwise alignment between β-strands using dynamic programming. For proteins with more than six β-strands, we first computed β-strand pairings using the BetaPro method. Then, we computed gapped alignments of the paired β-strands in parallel and anti-parallel directions and chose the interaction types and β-residue pairings with maximum alignment scores.

CHAPTER I

INTRODUCTION

Proteins are large, complex molecules made up of smaller subunits called amino acids. Chemical properties that distinguish the twenty amino acids cause the protein chains to fold up into specific three-dimensional structures that define their particular functions in the cell. There are four levels of protein structure. The primary structure refers simply to the "linear" sequence of amino acids. The secondary structure is the "locally" ordered structure that is created by hydrogen bonding within the protein backbone. The tertiary structure refers to the "global" folding of a single amino acid chain, and the quaternary structure involves the association of two or more chains into a multi-subunit structure.

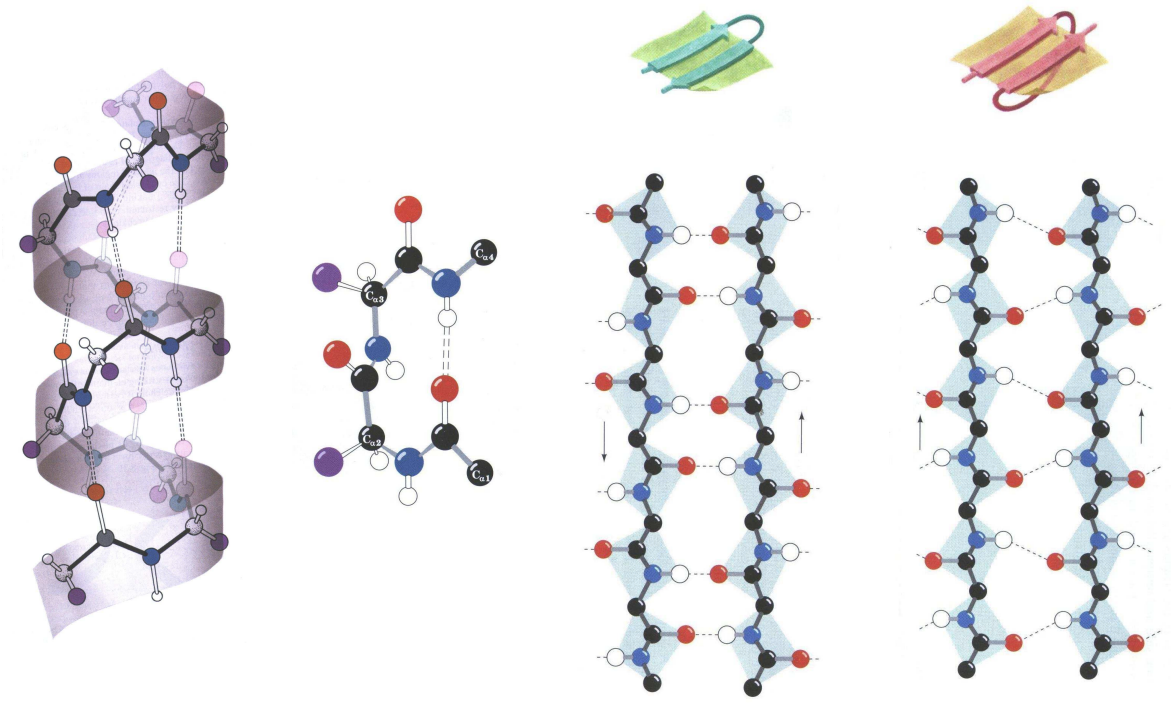
Protein structure prediction is one of the most fundamentally unsolved problems in computational molecular biology. There are several levels at which protein structure prediction can be performed. Secondary structure prediction is concerned with the assignment of each amino acid to a secondary structure state. In tertiary structure prediction (*e.g.,* protein folding), the goal is to predict the conformation assumed by a protein molecule in the three-dimensional (3-D) space. Tertiary structure prediction is important from many aspects. First of all, biological functions of proteins are dependent on their 3-D structure. Therefore, accurate prediction of the structure will provide information on the functional role of the protein. Second, protein structure prediction is an efficient alternative to experimental methods that solve structure, which are limited and time consuming. Third and most important, protein structure prediction enables us to design novel proteins and drugs, which is a fundamental task on the path toward curing diseases.

The prediction of the 3-D structure greatly benefits from the information related to secondary structure, solvent accessibility, and non-local contacts that stabilize a protein's structure. Therefore, the prediction of such components is vital to our understanding of the structure and function of a protein. In this thesis, we concentrate on the protein secondary structure prediction and the β-sheet prediction problems.

1.1 Protein Secondary Structure Prediction

The three major secondary structure states are the α-helix {H}, the β-strand {E}, and the loop {L}. α-helices are strengthened by hydrogen bonds between every fourth amino acid so that the protein backbone adopts a helical configuration, as shown in Figure 1(a). Likewise in loops, (*e.g.*, turns or bends), the hydrogen bonding is mostly local. For example, the turn segment in Figure 1(b) has a hydrogen bond between the oxygen and hydrogen atoms of the first and the fourth amino acids, respectively. The hydrogen bonding structure in β-strands is slightly different, where both local and non-local interactions are observed. In β-strands, the most common local hydrogen bonding is between every two amino acids, and non-local interactions are due to hydrogen bonds between amino acid pairs positioned in interacting β-strand segments. Those segments can adopt either a parallel or an anti-parallel conformation, as shown in Figure 1(c)-(d).

A protein secondary structure prediction algorithm assigns to each amino acid a structural state from a three-letter alphabet {H, E, L} representing the α-helix, β-strand, and loop, respectively. Protein secondary structure prediction is important as it provides insights into the functional role of a protein [144, 19, 57, 32, 70, 56, 145]. Prediction of function via sequence similarity search for new proteins (function annotation transfer) should be facilitated by a more accurate prediction of secondary



(a) An α-helix segment (b) A turn segment (c) Anti-parallel conformation (d) Parallel conformation

Figure 1: (a)-(b): Local interactions in α-helix and loop segments. (c)-(d): Non-local interactions in β-strand segments (Top diagrams illustrate β-strands in cartoon representation). In all diagrams, hydrogen bonds are shown as dashed lines. Solid lines represent covalent bonds. The color representations of the atoms in (a): Carbon (C_α)-dark gray, Carbon (in C=O group)-light gray, hydrogen-white, oxygen-red, nitrogen-blue. The color representations of the atoms in (b), (c) and (d): Carbon-black, hydrogen-white, oxygen-red, nitrogen-blue. Side-chains are represented as purple spheres. Reprinted from "Biochemistry, 3rd Edition", Donald Voet, Judith G. Voet, Copyright ©2004 John Wiley & Sons,Inc. Illustration, Irving Geis. Rights owned by Howard Hughes Medical Institute. Not to be used without permission.

structure since structure is more conserved than sequence. In addition, protein secondary structure prediction can be a step toward the prediction of the 3-D structure [36]. For instance, protein secondary structure information can be included into fold recognition methods, in which a target amino acid sequence with unknown structure is compared against a library of structural templates (folds) and the best scoring fold is assumed to be the one adopted by the sequence [83].

1.2 Protein Beta-Sheet Prediction

A β-sheet is a set of β-strand segments, which are involved in hydrogen bonding interactions. The association of β-sheets has been implicated in the formation of protein aggregates and fibrils observed in many human diseases, including Alzheimer's and mad cow diseases [85]. β-sheets can be open, meaning that they have two edge strands (as in the flavodoxin fold or the immunoglobulin fold) or they can be closed β-barrels (such as the TIM barrel). Open β-sheets are the most common sheet types observed in cellular proteins. An example is shown in Figure 2, where four β-strands interact pairwise to form an open β-sheet.

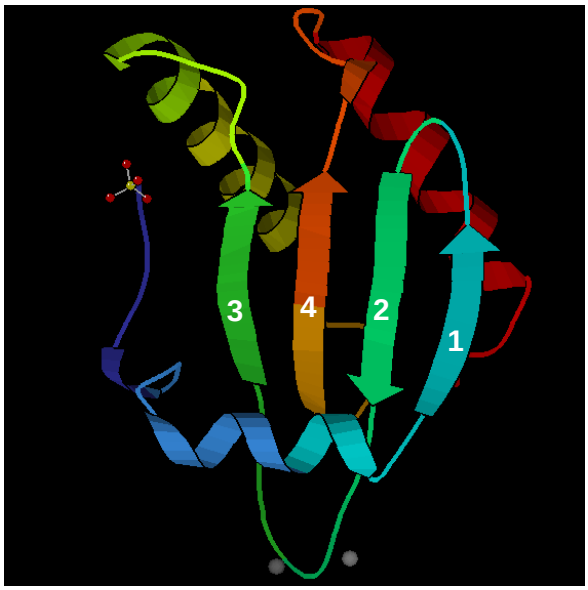
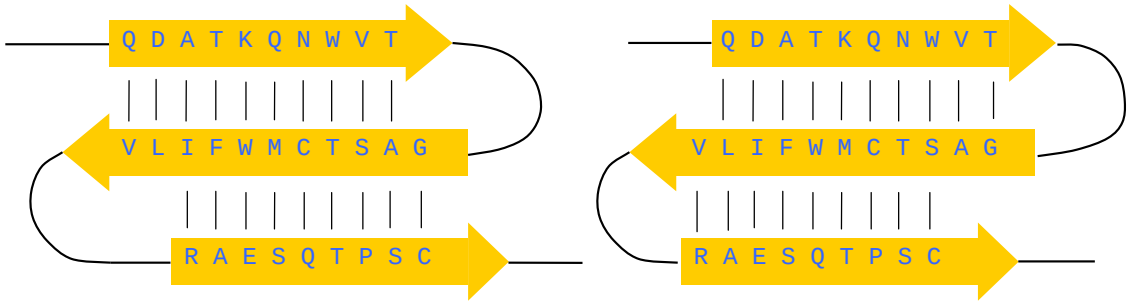


Figure 2: Secondary structure of the Rnase P protein (PDB id: 1A6F). β-strands that form the β-sheet are numbered in sequential order.

The conformational arrangement of β-strands that form β-sheets can be described by the following components: the assignment (or grouping) of β-strands into β-sheets, the spatial ordering of β-strand segments in each sheet, the interaction types of β-strand segment pairs, and amino acid residue interactions also known as contact maps. For instance, in Figure 2, four β-strands interact to form a single β-sheet. Here, the

β-strand segments are ordered as (1-2-4-3) in the spatial direction, in which the numbers represent the sequential indices of the β-strands[1]. The interaction types of the segments are such that the first and the second segments make an anti-parallel interaction, the second and the fourth segments make the second anti-parallel interaction, while the third and the fourth segments make a parallel interaction. As the fourth component of the β-sheet formation, a contact map defines the amino acid pairs that make non-local interactions (or residue pairs). In Figure 3, two possibilities are shown for the residue pairing pattern of a β-sheet with three β-strands. Both β-sheets have the same grouping, ordering and interaction type combination but their contact map is different.



(a) (b)

Figure 3: Two possibilities for the residue pairing pattern of a β-sheet with three β-strands. The letters represent the amino acids in β-strand segments. The vertical line segments show the amino acid interactions.

The β-sheet conformation of a protein is essential for understanding its structure [152]. Prediction of β-sheet conformation from amino acid sequence is useful, not only for predicting the tertiary structure, [150, 131] but also for elucidating folding pathways [97, 94] and designing new proteins [86, 88].

[1]For convenience, we start with the segment with smaller sequential index.

1.3 State-of-the-Art

Our goal in this section is to provide a brief overview of the prior work on protein secondary structure and β-sheet prediction. In the subsequent chapters, we will survey the related work about the problems under discussion in detail.

1.3.1 Protein Secondary Structure Prediction

Algorithms of protein secondary structure prediction frequently employ neural networks [80, 120, 115, 50, 96, 112, 80], support vector machines [66, 84, 146, 105, 104, 71] and hidden Markov models [134, 36, 20]. Parameters of the algorithm have to be defined by machine learning, therefore algorithm development and assessment usually contains four steps. First, a statistical analysis is performed to identify the most informative correlations and patterns. Then, a model is developed, which represents dependencies between structure and sequence elements. In the third step, the model parameters are derived from a training set. Finally, the algorithm prediction accuracy is assessed on test samples (sets) with known structure.

There are two types of algorithms in protein secondary structure prediction. A single-sequence algorithm does not utilize evolutionary information about other similar proteins. The algorithm should be suitable for a sequence with no similarity to any other protein sequence. Algorithms of another type are explicitly using sequences of related proteins, which often have similar structures. The prediction accuracy of such an algorithm should be higher than one of a single-sequence algorithm due to incorporation of additional evolutionary information from multiple alignments or multiple alignment profiles [128, 59]. The accuracy (sensitivity) of the best current single-sequence prediction methods is close to 70%. BSPSS [134], SIMPA [91], SOPM [62], and GOR IV [60] are examples of single-sequence prediction algorithms. Among the current best methods that use evolutionary information (multiple alignments or PSSM profiles), one can mention Porter [114], PSIPRED [80], SSpro [28],

APSSP2 [120], SVMpsi [84], PHDpsi [117], JPRED2 [50], and PROF [108]. The accuracy of the state-of-the-art algorithms that employ multiple alignments or alignment profiles is close to 80% [28]. For instance, the prediction accuracy of Porter was shown to be as high as 80.4% [8]. The secondary structure prediction performance can further be improved by consensus classifiers, in which different prediction methods are combined to improve over a single method [123, 65]. The joint utilization of methods that specialize on single-sequence prediction and methods using evolutionary information will definitely improve the prediction performance. The theoretical limit of the accuracy of secondary structure assignment from experimentally determined 3-D structure is estimated to be 88% [125]. A real-time analysis and comparison of various protein secondary structure prediction servers can be found at the EVAsec website [9]. A comprehensive evaluation of the protein secondary structure prediction algorithms can be found in Robles *et al.* [123].

1.3.2 Protein Beta-Sheet Prediction

Several methods have been proposed to understand and predict topological features of β-sheets. Methods that aim to improve our understanding of β-sheet formation analyzed the intrinsic and statistical propensities of amino acids [92, 98, 149, 153, 131], their evolutionary conservation [150, 94] and the contribution of these factors to local structure and β-sheet stability [148, 98, 138, 75]. Methods that predict β-strand interactions and/or amino acid residue contacts utilize statistical potentials [73, 74, 21, 153, 127], information theory [140] and machine learning [82, 29, 113, 67, 93, 39, 41, 118, 142, 30, 40]. Note that all these methods are developed for global proteins though similar ideas were also applied to predict contacts in specific folds [34] as well as transmembrane proteins that contain β-strand interactions [143, 121]. In this thesis, we are concentrating on globular proteins only.

Among the machine learning approaches, Cheng and Baldi [39] proposed BetaPro,

which is a three-stage modular approach that predicts and assembles the β-sheets of a native protein. BetaPro utilizes recursive neural networks followed by dynamic programming and graph theory to exploit global covariation and constraints characteristic of β-sheets. To derive the residue interaction propensities, BetaPro utilizes information from 10 surrounding residues instead of modeling each pair as independent. BetaPro has 68% sensitivity and 61% positive predictive value (PPV) in the segment pairing category when true secondary structure and solvent accessibility information is used, which is a significant improvement over statistical data-driven approaches. BetaPro was followed by SVMcon, a new contact map predictor that uses support vector machines to predict medium- and long-range contacts [40]. Recently, Jeong et al. [79] investigated two new algorithms for predicting β-strand partners. The first algorithm poses the problem as integer linear programming optimization problem and solves it using the ILOG CPLEX^{TM} package. The second approach is greedy and it explicitly encourages two simple folding rules.

1.4 Contributions of the Thesis

In this thesis, we develop Bayesian models and algorithms for protein secondary structure and β-sheet prediction. In secondary structure prediction, we make the following contributions:

- We derive a Bayesian framework for proteins in the single-sequence setting.
- We extract the most informative correlations between amino acid pairs in each secondary structure type (feature sets) by performing a χ^2-test.
- We derive probability models for the observation of amino acid residues. Each model specializes on a different section of the dependency structure and considers dependencies to forward and/or backward positions as well as dependencies to positions outside of a segment.

- We develop a hidden semi-Markov model (HSMM) for each amino acid observation model. We combine the HSMMs by taking the average of the marginal posterior probabilities.

- We develop training set reduction methods to refine estimates of the HSMM parameters.

- We develop N-best algorithms to compute suboptimal segmentations of secondary structure.

- We develop a non-local interaction model for β-sheets and incorporate it into the single-sequence prediction method using the N-best approach.

In β-sheet prediction, our contributions can be listed as follows:

- We introduce a Bayesian framework for proteins with six or less β-strands, in which we model the conformational features in a probabilistic framework by combining the amino acid pairing potentials with *a priori* knowledge of β-strand organizations.

- We develop efficient heuristics to compute the optimum β-sheet architecture.

- We develop a dynamic programming algorithm to find the optimum pairwise alignment between β-strands. Allowing any number of gaps in an alignment enables us to model β-bulges.

1.5 Organization of the Thesis

Chapter 2 presents our work on protein secondary structure prediction in the single-sequence setting. It includes a Bayesian formulation, a statistical analysis, feature sets, dependency models, the hidden semi-Markov model implementation, and training set reduction methods. Several simulation results are provided to demonstrate the effectiveness of the proposed approaches.

Chapter 3 investigates the feasibility of incorporating non-local interactions into the single-sequence prediction method we developed in Chapter 2. It presents two N-best decoding algorithms and a Bayesian β-sheet model. Simulation results are included for the non-local interaction model and potential extensions are discussed.

Chapter 4 explains our work on beta-sheet prediction. A Bayesian framework is introduced for proteins with six or less β-strands. For an efficient computation of the optimum conformation, heuristic and dynamic programming algorithms are developed. Several simulation results are presented to show the effectiveness of the proposed method.

Chapter 5 concludes the thesis by discussing future directions.

CHAPTER II

PROTEIN SECONDARY STRUCTURE PREDICTION FOR A SINGLE-SEQUENCE WITH HIDDEN SEMI-MARKOV MODELS

2.1 Introduction

The accuracy of protein secondary structure prediction has been improving steadily towards the 88% estimated theoretical limit [125]. There are two types of prediction algorithms: Single-sequence algorithms imply that evolutionary information about other related proteins is not available, while algorithms of the second type imply that this information is available, and use it intensively. The single-sequence algorithms could make an important contribution to studies of proteins with no detected relatives, however the accuracy of protein secondary structure prediction from a single-sequence is not as high as when the additional evolutionary information is present.

Single-sequence algorithms for protein secondary structure prediction are important because a significant percentage of the proteins identified in genome sequencing projects have no detectable sequence similarity to any known protein [141, 100]. Particularly in sequenced prokaryotic genomes, about a third of the protein coding genes are annotated as encoding hypothetical proteins lacking similarity to any protein with a known function [78]. Also, out of the 25,000 genes believed to be present in the human genome, no more than 40-60% can be assigned a functional role based on similarity to known proteins [77, 48]. For a larger picture, the Pfam database allows one to get information on distribution of proteins with known functional domains in three domains of life (see Table 1). From the structure prediction standpoint it is important that two or more hypothetical proteins may bear similarity with each other,

Table 1: Number of proteins with known functional domains.

	# Proteins	# Proteins with Pfam hit	(%)
Bacteria	623,037	450,962	72.38
Archaea	50,406	33,259	65.98
Eukaryota	284,392	187,472	65.92
Total	957,835	671,693	70.13

in which case it would still be possible to incorporate evolutionary information in a structure prediction algorithm. However, many hypothetical proteins would not have detectable similarity to any protein at all. Such "orphan" proteins may represent a sizeable portion of a proteome[1] as shown in Table 2. For an orphan protein, any

Table 2: Statistics of hypothetical proteins and orphan proteins observed in the recently sequenced genomes (year 2004).

	# Proteins	(%) hypothetical proteins	(%) orphans in hypotheticals
Sulfolobus islandicus (Archaea)	197	65.98	57.69
Bacillus claussi (Bacteria)	4121	31.64	18.66
Gallus gallus (Eukaryota)	29,172	11.84	32.4

method of secondary structure prediction performs as a single-sequence method. Developing better methods of secondary structure prediction from single-sequence has a definite merit as it helps improving the functional annotation of orphan proteins.

In this chapter, we introduce a new method for protein secondary structure prediction, which develops further the model proposed by Schmidler *et al.* [134]. We formulate the problem in a Bayesian framework, which enables us to implement hidden semi-Markov models. For dimensionality reduction, we performed a statistical analysis and identified the most informative correlations between sequence and structure variables. We specifically considered correlations at proximal positions of structural segments and dependencies to upstream and downstream residues. In addition,

[1]Proteome is the complete set of proteins that can be expressed by the genetic material of an organism.

we developed training set reduction methods to refine estimates of the HSMM parameters. The three-state-per-residue accuracy and other accuracy measures of the proposed method, IPSSP, are shown to be comparable or better than the state-of-the-art methods tested under the single-sequence condition.

2.2 Model Derivation

2.2.1 Bayesian Formulation

The linear sequence that defines a secondary structure of a protein can be described by the pair $(\mathbf{S}, \mathbf{T})$, where $\mathbf{S}$ is a sequence of the structural segment end (border) positions and $\mathbf{T}$ is a sequence that determines the structural state of each segment (α-helix, β-strand, or loop). For instance, for the secondary structure shown in Figure 4, $\mathbf{S}$ = (4, 9, 12, 16, 21, 28, 33) and $\mathbf{T}$ = (L, E, L, E, L, H, L).

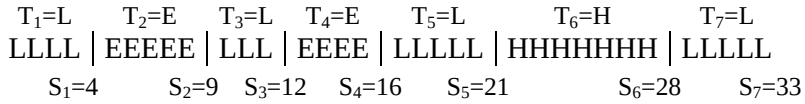


Figure 4: The secondary structure segmentation and its representation by structural segments.

Given a statistical model specifying probabilistic dependencies between sequence and structure elements, the problem of protein secondary structure prediction could be stated as the problem of maximizing the *a posteriori* probability of a structure given the primary sequence. Thus, given the sequence of amino acids, $\mathbf{R}$ [2], one has to find the pair $(\mathbf{S}, \mathbf{T})$ that maximizes the *a posteriori* probability $P(\mathbf{S}, \mathbf{T} \mid \mathbf{R})$ defined by an appropriate statistical model. Using Bayes' rule, this probability can be expressed as

$$P(\mathbf{S}, \mathbf{T} \mid \mathbf{R}) = \frac{P(\mathbf{R} \mid \mathbf{S}, \mathbf{T}) P(\mathbf{S}, \mathbf{T})}{P(\mathbf{R})}, \tag{1}$$

where $P(\mathbf{R} \mid \mathbf{S}, \mathbf{T})$ denotes the likelihood and $P(\mathbf{S}, \mathbf{T})$ is the *a priori* probability. Since $P(\mathbf{R})$ is constant with respect to $(\mathbf{S}, \mathbf{T})$, maximizing $P(\mathbf{S}, \mathbf{T} \mid \mathbf{R})$ is equivalent

[2] $\mathbf{R} = (R_1, .., R_n)$, where R_i is the i^{th} amino acid.

to maximizing $P(\mathbf{R} \mid \mathbf{S}, \mathbf{T})P(\mathbf{S}, \mathbf{T})$. Hence, the MAP estimator takes the following form:

$$(\mathbf{S}, \mathbf{T})_{MAP} = \arg\max_{(\mathbf{S},\mathbf{T})} P(\mathbf{R} \mid \mathbf{S}, \mathbf{T})P(\mathbf{S}, \mathbf{T}). \tag{2}$$

To proceed further, we need models for each of these probabilistic terms. We model the *a priori* probability $P(\mathbf{S}, \mathbf{T})$ as follows:

$$P(\mathbf{S}, \mathbf{T}) = \prod_{j=1}^{m} P(T_j \mid T_{j-1})P(S_j \mid S_{j-1}, T_j), \tag{3}$$

where, m denotes the total number of secondary structure segments and $P(T_j \mid T_{j-1})$ is the probability of transition from a segment of secondary structure type T_{j-1} to a segment of secondary structure type T_j. Table 3 shows the transition probabilities $P(T_j \mid T_{j-1})$, estimated from the PDB_SELECT dataset (see Section 2.5.1.1). The

Table 3: The matrix of transition probabilities, $P(T_j \mid T_{j-1})$, used in the hidden semi-Markov model. Rows represent T_{j-1} values.

$P(T_j \mid T_{j-1})$	H	E	L
H	——	0.031	0.969
E	0.029	——	0.971
L	0.314	0.686	——

third term, $P(S_j \mid S_{j-1}, T_j)$, reflects the length distribution of a secondary structure segment. We can assume that

$$P(S_j \mid S_{j-1}, T_j) = P(S_j - S_{j-1} \mid T_j), \tag{4}$$

where $S_j - S_{j-1}$ is equal to the segment length (see Figure 4). The typical form of the segment length distribution for different secondary structure types is illustrated in [44, 134, 1].

The likelihood term $P(\mathbf{R} \mid \mathbf{S}, \mathbf{T})$ can be written as

$$\begin{aligned} P(\mathbf{R} \mid \mathbf{S}, \mathbf{T}) &= \prod_{j=1}^{m} P(\mathbf{R}_{[S_{j-1}+1:S_j]} \mid \mathbf{S}, \mathbf{T}) \\ &= \prod_{j=1}^{m} P(\mathbf{R}_{[S_{j-1}+1:S_j]} \mid S_{j-1}, S_j, T_j), \end{aligned} \tag{5}$$

where $\mathbf{R}_{[p:q]}$ denotes the sequence of amino acid residues with position indices from p to q. The probability of observing a particular amino acid sequence in a segment adopting a particular type of secondary structure is $P(\mathbf{R}_{[S_{j-1}+1:S_j]} \mid \mathbf{S}, \mathbf{T})$. This term is assumed to be equal to $P(\mathbf{R}_{[S_{j-1}+1:S_j]} \mid S_{j-1}, S_j, T_j)$. Thus, this probability depends only on the secondary structure type of a given segment, and not of adjacent segments. Note that we ignore the non-local interactions observed in β-sheets. This simplification allows us to implement an efficient hidden semi-Markov model.

To elaborate on the segment likelihood terms in Eq. (5), we have to consider the correlation patterns within a secondary structure segment. These patterns reflect the secondary structure specific physico-chemical interactions. For instance, α-helices are strengthened by hydrogen bonds between amino acid pairs situated at specific distances. To correctly define the likelihood term, we should also pay attention to the proximal positions, typically the four initial and the four final positions of a segment. In particular, α-helices include capping boxes, where the hydrogen bonding patterns and side-chain interactions are different from the internal positions [22, 54]. The observed distributions of amino acid frequencies in proximal (capping boxes) and internal positions of α-helix segments are depicted in Schmidler *et al.* [134], and show noticeably distinct patterns.

Presence of this inhomogeneity in the statistical model leads to the following expression for $P(\mathbf{R}_{[S_{j-1}+1:S_j]} \mid S_j, S_{j-1}, T_j)$:

$$\begin{aligned} P(\mathbf{R}_{[S_{j-1}+1:S_j]} \mid S_j, S_{j-1}, T_j) &= P_{N_1}(R_{k_b+1}) \prod_{i=k_b+2}^{l_N+k_b} P_{N_{i-k_b}}(R_i \mid R_{i-1}, .., R_{k_b+1}) \\ &\times \prod_{i=l_N+k_b+1}^{k_n-l_C} P_{Int}(R_i \mid R_{i-1}, .., R_{k_b+1}) \\ &\times \prod_{i=-l_C+1}^{0} P_{C_{1-i}}(R_{i+k_n} \mid R_{i+k_n-1}, .., R_{k_b+1}). \end{aligned} \quad (6)$$

Here, the first and the third sub-products represent the probability of observing l_N and l_C specific amino acids at the segment's N-terminal and C-terminal, respectively. The

second sub-product defines the observation probability of amino acids in the segment's internal positions. Note that k_b and k_e designate $S_{j-1}+1$ and S_j, respectively. The probabilistic expression in Eq. (6) is generic for α-helices, β-strands, and loops. Eq. (6) assumes that the probabilistic model is fully dependent within a segment, *i.e.*, observation of an amino acid at a particular position of a segment depends on all previous amino acids within that segment. However, at this time, the Protein Data Bank (PDB [15]) does not have a sufficient amount of experimental data to reliably estimate all the parameters of a fully dependent model. Therefore, it is important to reduce the dependency structure and keep only the most significant correlations. In order to achieve this goal, we performed the statistical analysis described in the following section.

2.2.2 Correlation Patterns of Amino Acids

Amino acids have distinct propensities for the adoption of secondary structure conformations [51]. These propensities are in the heart of many secondary structure prediction methods [43, 54, 22, 122, 116, 52, 46, 47, 90, 111]. Our goal is to come up with a dependency pattern that is comprehensive enough to capture the essential correlations yet simple enough in terms of the number of model parameters to allow reliable parameter estimation from the available training data. With this motivation, we performed a χ^2-test to identify the most significant correlations between amino acid pairs located in adjacent and non-adjacent positions for each type of secondary structure segments.

2.2.2.1 χ^2-Test

A χ^2-test is a statistical hypothesis test in which the test statistic has a χ^2 distribution when the null hypothesis is true or the probability distribution of the test statistic (assuming the null hypothesis is true) can be made to approximate a χ^2 distribution as closely as desired by making the sample size large enough. The χ^2-test is used in

two similar but distinct circumstances: (1) for estimating how closely an observed distribution matches an expected distribution. This is also referred as the goodness-of-fit test; (2) for estimating whether two random variables are independent. In our case, we are interested in the second category, in which we derive the degree of correlation between the amino acid pairs situated at various positions.

The first step of a χ^2-test is to establish hypotheses. Let (R_i, R_j) be an amino acid pair, where R_i is the amino acid at position i and R_j is the amino acid at position j. The null hypothesis claims that the two amino acids are independent whereas the alternative hypothesis states that the amino acids are correlated. The key idea of a χ^2-test is the comparison of observed and expected values. This is summarized in the following test statistic:

$$\chi^2 = \sum_{ij} \frac{O_{ij} - E_{ij}}{E_{ij}}, \tag{7}$$

where O_{ij} and E_{ij} are the observed and expected values of the amino acid pair (R_i, R_j), respectively, and χ^2 is the test statistic. When the test statistic is greater than the statistical significance threshold, we reject the null hypothesis and conclude that the amino acids are correlated. Otherwise, the pair is assumed to be independent. In statistics, parameters of a χ^2-test are typically recorded in contingency tables. Since an amino acid can take twenty possible values, a contingency table of size 20×20 can be used to analyze the correlations between amino acid pairs. In that case, the threshold would be 404.6 for a statistical significance level of 0.05.

2.2.2.2 Correlations within Segments

We first performed a χ^2-test and compared the empirical distribution of an amino acid pair with the respective product of marginal distributions. Therefore, we computed a 20×20 contingency table, which includes the frequencies of possible amino acid pairs observed in different structural states. We first analyzed the correlations between amino acid pairs at various separation distances. As highlighted in Table 4, we found

that in a α-helix segment, a residue at position i is highly correlated with residues at positions $i-2$, $i-3$ and $i-4$. Similarly, a β-strand residue has its highest correlations with residues at positions $i-1$, $i-2$, and a loop residue has its most significant correlation with a residue at position $i-1$. The test statistics for the remaining pairs are above the statistical significance threshold but these values are considerably lower than the ones highlighted in Table 4. The dependencies identified by the statistical analysis are in agreement with the well known physical nature of the secondary structure conformations.

Table 4: Correlations at the amino acid level as characterized by the χ^2 measure (PDB_SELECT set).

	Helix		Strand		Loop	
Separation	χ^2	# of pairs	χ^2	# of pairs	χ^2	# of pairs
1	1854.34	118,324	**2579.85**	60,423	**9600.85**	154,404
2	**7008.83**	103,853	**1832.78**	44,121	5774.58	124,249
3	**2454.03**	89,414	1116.65	30,909	4828.13	100,325
4	**5095.27**	77,302	535.02	20,336	2276.21	80,930
5	2052.68	67,036	461.70	12,584	1298.16	66,109
6	1295.46	57,602	398.44	7361	950.66	54,993
7	2196.94	49,017	392.93	4196	895.42	46,391
8	627.00	41,350	355.81	2292	761.48	39,611

2.2.2.3 Position Specific Correlations

In this section, we derived position specific correlations within each type of secondary structure segment. We analyzed proximal positions and a representative set of internal positions. Frequency patterns in proximal positions deviate from the patterns observed in internal positions [22, 52]. For a better quantification, we first computed the Kullback-Liebler (KL) distance between the probability distributions of the proximal and the internal positions as shown in Table 5. From this table, we can observe that the KL distance is significantly higher for positions closer to segment borders. This shows that amino acids in proximal locations have significantly different distributions from those at internal regions. After making this observation, we performed

Table 5: KL distance between distributions of amino acids in proximal and internal positions (PDB_SELECT set).

KL-dis	N1	N2	N3	N4	C4	C3	C2	C1
α-Helix	**0.402**	**0.194**	**0.100**	**0.053**	0.018	0.018	0.020	**0.036**
β-Strand	**0.047**	**0.025**	0.019	——	——	0.021	**0.039**	**0.074**
Loop	**0.045**	0.019	0.008	0.003	0.004	0.008	**0.026**	**0.028**

a χ^2-test for proximal positions to identify the correlations between amino acid pairs at various separation distances. Tables 6, 7, and 8 summarize the results for α-helix, β-strand, and loop segments, respectively. From these tables, we can see that the general assumption of segment independence does not hold because there is a significant correlation between residues situated on both sides of the segment borders. For instance, in Table 6 the amino acid at position i = N2, significantly correlates with the amino acid at position $i-2$, which is outside the segment. This correlation can be caused by physical interactions between nearby residues [22]. Also, the strength of correlation for the $i+(downstream)$ residues at the C-terminal border is different from the strength observed for $i-(upstream)$ residues at the N-terminal border. This fact indicates an asymmetry in the correlation behavior for $i+$ and $i-$ residues.

Table 6: Position specific correlations as characterized by the χ^2 measure in the proximal positions of α-helices (PDB_SELECT set).

χ^2	N1	N2	N3	N4	C4	C3	C2	C1
$i-5$	380.33	410.29	421.77	538.17	604.79	628.98	549.77	563.37
$i-4$	491.35	409.30	591.87	482.28	**830.18**	**963.03**	**1261.04**	**1213.12**
$i-3$	416.40	637.47	**2000.33**	552.22	696.89	632.99	624.90	**714.48**
$i-2$	524.46	**1029.24**	**731.30**	**649.11**	**1082.25**	**1181.51**	**1270.42**	**1300.46**
$i-1$	**708.76**	**770.43**	661.04	614.21	481.05	497.00	603.44	810.35
$i+1$	**770.43**	**805.38**	**702.25**	470.58	463.31	527.86	**717.04**	**1266.49**
$i+2$	**982.18**	**993.92**	**844.97**	**827.26**	**933.17**	**903.81**	591.22	631.45
$i+3$	**875.59**	619.44	697.18	465.17	578.54	485.95	507.21	482.92
$i+4$	**1132.54**	**872.68**	694.83	**1055.63**	657.20	443.62	397.44	476.19
$i+5$	487.41	594.32	652.90	468.99	527.83	370.97	378.79	454.61

A similar asymmetry in the correlation pattern was also observed for internal positions as shown in Table 9. For instance, in an α-helix segment, typically, the i^{th} residue at an internal position is highly correlated with the $i-2^{th}$, $i-3^{th}$, $i-4^{th}$,

Table 7: Position specific correlations as characterized by the χ^2 measure in the proximal positions of β-strands (PDB_SELECT set).

χ^2	N1	N2	N3	N4	C4	C3	C2	C1
$i-5$	465.37	403.31	413.26	395.04	352.55	330.02	378.60	376.07
$i-4$	654.98	456.38	483.51	441.46	400.92	390.53	474.38	693.12
$i-3$	510.14	519.55	428.88	428.16	422.03	491.73	510.24	653.66
$i-2$	**897.90**	654.51	**601.60**	**565.55**	**492.39**	**543.82**	622.69	529.75
$i-1$	**948.59**	**853.73**	574.88	445.42	462.32	510.58	**787.74**	**898.29**
$i+1$	**1040.43**	**741.91**	496.39	**499.59**	415.23	**598.74**	**688.16**	573.46
$i+2$	**717.47**	**842.68**	**572.70**	489.51	391.72	438.24	536.08	**626.48**
$i+3$	544.38	504.97	495.69	398.30	400.94	399.77	558.58	594.21
$i+4$	576.12	496.33	403.25	403.55	352.35	425.08	436.07	571.56
$i+5$	394.77	446.47	364.07	373.41	358.11	378.73	595.02	372.85

Table 8: Position specific correlations as characterized by the χ^2 measure in the proximal positions of loops (PDB_SELECT set).

χ^2	N1	N2	N3	N4	C4	C3	C2	C1
$i-5$	525.52	438.41	440.73	360.78	436.80	373.60	476.23	573.17
$i-4$	**821.26**	706.34	483.41	367.51	375.90	390.95	507.84	594.84
$i-3$	**897.63**	513.38	**628.45**	374.36	414.06	473.22	486.77	496.24
$i-2$	**1071.32**	651.56	529.75	370.03	**499.76**	449.44	572.50	797.24
$i-1$	**1123.73**	**1069.17**	618.63	**470.26**	399.52	**560.42**	**1180.19**	**944.17**
$i+1$	**1163.89**	**977.29**	**733.89**	419.78	**469.27**	**634.31**	721.94	**1145.00**
$i+2$	685.76	580.68	551.32	365.14	440.72	578.76	**789.16**	694.47
$i+3$	**916.21**	631.75	438.45	435.29	395.91	620.07	**945.19**	635.66
$i+4$	655.16	498.59	502.03	401.33	379.38	512.19	633.13	643.62
$i+5$	457.12	362.01	407.11	356.63	389.33	433.25	552.84	483.10

$i-5^{th}$, $i+2^{th}$, and the $i+4^{th}$ residues. The correlation strength between the i^{th} residue and the $i-2^{th}$ residue is different from the one observed for the i^{th} and the $i+2^{th}$ residues.

In the next section, we will refine the probabilistic model needed to determine $P(\mathbf{R}_{\,[S_{j-1}+1:S_j]} \mid S_{j-1}, S_j, T_j)$ using the most significant correlations identified by the statistical analysis.

2.2.3 Reduced Dependency Model

Correlation analysis allows us to reduce the alphabet size in Eq. (6) by selecting only the most significant correlations. The dependence patterns revealed by the statistical analysis are shown in Table 10 divided into panels for α-helix (H), β-strand (E), and loop (L) structures. To reduce the dimension of the parameter space, we grouped

Table 9: Position specific correlations as characterized by the χ^2 measure in internal positions of α-helices, β-strands, and loops (PDB_SELECT set).

χ^2	α-Helix	β-Strand	Loop
$i-5$	**1654.88**	479.43	441.17
$i-4$	**3797.61**	426.88	514.73
$i-3$	**1839.16**	**527.62**	**637.94**
$i-2$	**3445.50**	**601.78**	**592.88**
$i-1$	1006.62	**540.03**	**642.02**
$i+1$	821.42	**409.86**	**526.12**
$i+2$	**1883.89**	382.90	**488.29**
$i+3$	891.31	370.13	**549.43**
$i+4$	**1210.99**	320.33	389.41
$i+5$	587.33	299.03	389.12

the amino acids into three or five hydrophobicity classes. We used five classes only for positions that have significantly high correlation measures. In Table 10, h^3_{i-1} stands for the dependency of an amino acid at position i to the hydrophobicity class of an amino acid at position $i-1$, and the superscript 3 represents the number of hydrophobicity classes used. To fully utilize the dependency structure, we found it useful to derive three separate dependency models. The first model, $\mathcal{M}1$, utilizes only dependencies to upstream positions, $(i-)$, the second model, $\mathcal{M}2$, includes dependencies to upstream $(i-)$, and downstream $(i+)$ positions simultaneously, and the third model, $\mathcal{M}3$, incorporates only downstream $(i+)$ dependencies. In our model, we distinguished positions within a segment as proximal and internal. We identified as proximal positions those in which the amino acid frequency distributions significantly deviate from the ones in internal positions in terms of the KL distance (see Table 5). Based on the available training data, we chose 6 proximal positions (N1-N4, C1-C2) for α-helices, 4 proximal positions (N1-N2, C1-C2) for β-strands, and 8 proximal positions (N1-N4, C1-C4) for loops. The remaining positions are defined as internal positions (Int).

In addition to position specific dependencies, we derived separate patterns for segments with different lengths. Table 10 shows the dependence patterns for segments longer than L residues, where L is five for α-helices, four for β-strands and three for

Table 10: Positional dependencies within structural segments for the models $\mathcal{M}1$, $\mathcal{M}2$, and $\mathcal{M}3$. Segments longer than L residues are considered. $h_j^3 \in \{hydrophobic, neutral, hydrohilic\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$. h_j^5 is a five letter alphabet with groups defined as $\{P, G\}$, $\{E, K, R, Q\}$, $\{D, S, N, T, H, C\}$, $\{I, V, W, Y, F\}$, $\{A, L, M\}$.

		$\mathcal{M}1$	$\mathcal{M}2$	$\mathcal{M}3$
H	Int	h_{i-2}^5, h_{i-3}^3, h_{i-4}^5, h_{i-7}^3	h_{i-2}^3, h_{i-3}^3, h_{i-4}^3, h_{i+2}^3, h_{i+4}^3	h_{i+2}^5, h_{i+3}^5, h_{i+4}^5
	N1	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+2}^5	h_{i+2}^5, h_{i+4}^5
	N2	h_{i-1}^3, h_{i-2}^3, h_{i-3}^3	h_{i-2}^3, h_{i+2}^3, h_{i+4}^3	h_{i+1}^3, h_{i+2}^3, h_{i+4}^3
	N3	h_{i-1}^3, h_{i-2}^3, h_{i-3}^3	h_{i-2}^3, h_{i-3}^3, h_{i+2}^3	h_{i+1}^3, h_{i+2}^3, h_{i+4}^3
	N4	h_{i-1}^3, h_{i-2}^3, h_{i-3}^3	h_{i-1}^3, h_{i+2}^3, h_{i+4}^3	h_{i+1}^3, h_{i+2}^3, h_{i+4}^3
	C1	h_{i-1}^3, h_{i-2}^3, h_{i-4}^3	h_{i-2}^3, h_{i-4}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3, h_{i+3}^3
	C2	h_{i-2}^3, h_{i-3}^3, h_{i-4}^3	h_{i-2}^3, h_{i-4}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3, h_{i+3}^3
E	Int	h_{i-1}^5, h_{i-2}^5, h_{i-3}^3	h_{i-1}^3, h_{i-2}^3, h_{i+1}^3, h_{i+2}^3	h_{i+1}^5, h_{i+2}^5, h_{i+3}^3
	N1	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+1}^5	h_{i+1}^5, h_{i+2}^5
	N2	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+2}^5	h_{i+1}^5, h_{i+2}^5
	C1	h_{i-1}^3, h_{i-3}^3, h_{i-4}^3	h_{i-1}^3, h_{i-3}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3, h_{i+3}^3
	C2	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+1}^5	h_{i+1}^5, h_{i+2}^5
L	Int	h_{i-1}^5, h_{i-2}^5, h_{i-3}^3, h_{i-4}^3	h_{i-1}^5, h_{i-2}^3, h_{i+1}^5, h_{i+2}^3	h_{i+1}^5, h_{i+2}^5, h_{i+3}^3, h_{i+4}^3
	N1	h_{i-1}^5, h_{i-2}^5, h_{i-3}^3	h_{i-1}^5, h_{i-2}^5, h_{i+1}^3	h_{i+1}^5, h_{i+2}^5, h_{i+3}^3
	N2	h_{i-1}^5, h_{i-2}^3, h_{i-4}^3	h_{i-1}^5, h_{i-2}^3, h_{i+1}^3	h_{i+1}^5, h_{i+2}^3, h_{i+4}^3
	N3	h_{i-1}^5, h_{i-2}^3, h_{i-3}^3	h_{i-1}^5, h_{i-2}^3, h_{i+1}^3	h_{i+1}^5, h_{i+2}^3, h_{i+3}^3
	N4	h_{i-1}^5, h_{i-2}^3, h_{i-3}^3	h_{i-1}^5, h_{i-2}^3, h_{i+1}^3	h_{i+1}^5, h_{i+2}^3, h_{i+3}^3
	C1	h_{i-1}^5, h_{i-2}^5, h_{i-3}^3	h_{i-1}^5, h_{i-2}^5, h_{i+1}^3	h_{i+1}^5, h_{i+2}^5, h_{i+3}^3
	C2	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+3}^5	h_{i+1}^5, h_{i+2}^5
	C3	h_{i-1}^3, h_{i-2}^3, h_{i-3}^3	h_{i-1}^3, h_{i+1}^3, h_{i+2}^3	h_{i+1}^3, h_{i+2}^3, h_{i+3}^3
	C4	h_{i-1}^3, h_{i-2}^3, h_{i-3}^3	h_{i-2}^3, h_{i-3}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3, h_{i+3}^3

loops. For shorter segments, we selected a representative set of patterns from Table 10 according to the available training data (see Tables 11, 12, and 13).

For positions close to the sequence ends, *i.e.*, the first five (N-terminal) and the last five (C-terminal) amino acids of the protein, we excluded the dependencies that fall outside the amino acid sequence. For instance, for the first five amino acids, we excluded the $i-$ dependencies and for the last five amino acids, we excluded the $i+$ dependencies that fall outside the protein. Tables 14 and 15 show the dependency sets for the amino acids at sequence ends.

Table 11: Positional dependencies within α-helix segments for the models $\mathcal{M}1$, $\mathcal{M}2$, and $\mathcal{M}3$. Segments with L or less residues are considered. l is the segment length. $h_j^3 \in \{hydrophobic, neutral, hydrohilic\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$. h_j^5 is a five letter alphabet with groups defined as $\{P, G\}$, $\{E, K, R, Q\}$, $\{D, S, N, T, H, C\}$, $\{I, V, W, Y, F\}$, $\{A, L, M\}$.

α-helix		$\mathcal{M}1$	$\mathcal{M}2$	$\mathcal{M}3$
l=3	N1	h^3_{i-1}, h^3_{i-2}	h^3_{i-1}, h^3_{i+2}	h^3_{i+2}, h^3_{i+4}
	N2	h^3_{i-1}, h^3_{i-2}	h^3_{i-2}, h^3_{i+2}	h^3_{i+2}, h^3_{i+4}
	C1	h^3_{i-2}, h^3_{i-4}	h^3_{i-2}, h^3_{i+1}	h^3_{i+1}, h^3_{i+2}
l=4	N1	h^3_{i-1}, h^3_{i-2}	h^3_{i-1}, h^3_{i+2}	h^3_{i+2}, h^3_{i+4}
	N2	h^3_{i-1}, h^3_{i-2}	h^3_{i-2}, h^3_{i+2}	h^3_{i+2}, h^3_{i+4}
	C1	h^3_{i-2}, h^3_{i-4}	h^3_{i-2}, h^3_{i+1}	h^3_{i+1}, h^3_{i+2}
	C2	h^3_{i-2}, h^3_{i-4}	h^3_{i-2}, h^3_{i+1}	h^3_{i+1}, h^3_{i+2}
l=5	N1	h^3_{i-1}	h^3_{i-1}	h^3_{i+2}
	N2	h^3_{i-2}	h^3_{i-2}	h^3_{i+2}
	N3	h^3_{i-3}	h^3_{i-3}	h^3_{i+2}
	C1	h^3_{i-2}	h^3_{i-2}	h^3_{i+1}
	C2	h^3_{i-2}	h^3_{i-2}	h^3_{i+1}

For each dependency model ($\mathcal{M}1$-$\mathcal{M}3$), the probability of observing an amino acid at a given position is defined using the dependence patterns selected from Table 10. For instance, according to the model $\mathcal{M}2$, the conditional probability of observing an amino acid at position i=N3 of an α-helix segment becomes $P_{N_3}(R_i \mid h^3_{i-2}, h^3_{i-3}, h^3_{i+2})$. As formulated in Eq. (6), we multiply conditional probabilities and obtain the propensity of observing an amino acid segment given a secondary structure type and a model. In the case of $\mathcal{M}1$, and $\mathcal{M}3$, this product gives the segment likelihood expression, which is a properly normalized probability value $P(\mathbf{R}_{\,[S_{j-1}+1:S_j]} \mid \mathbf{S}, \mathbf{T})$. Hence, $\mathcal{M}1$, and $\mathcal{M}3$ are probabilistic models. For $\mathcal{M}2$, we rather obtain a score $Q(\mathbf{R}_{\,[S_{j-1}+1:S_j]} | \mathbf{S}, \mathbf{T})$ that represents the potential of an amino acid segment to adopt a particular secondary structure conformation. This scoring system can be used to characterize amino acid segments in terms of their propensities to form structures of different types and when uniformly applied to compute segment potentials, allows us

Table 12: Positional dependencies within β-strand segments for the models $\mathcal{M}1$, $\mathcal{M}2$, and $\mathcal{M}3$. Segments with L or less residues are considered. l is the segment length. $h_j^3 \in \{hydrophobic, neutral, hydrohilic\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$. h_j^5 is a five letter alphabet with groups defined as $\{P, G\}$, $\{E, K, R, Q\}$, $\{D, S, N, T, H, C\}$, $\{I, V, W, Y, F\}$, $\{A, L, M\}$.

β-strand		$\mathcal{M}1$	$\mathcal{M}2$	$\mathcal{M}3$
l=1	N1	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+1}^5	h_{i+1}^5, h_{i+2}^5
l=2	N1	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	C1	h_{i-1}^3, h_{i-3}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+3}^3
l=3	N1	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	N2	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+2}^3	h_{i+1}^3, h_{i+2}^3
	C1	h_{i-1}^3, h_{i-3}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
l=4	N1	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	N2	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+2}^3	h_{i+1}^3, h_{i+2}^3
	C1	h_{i-1}^3, h_{i-3}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	C2	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3

to implement algorithms following the theory of hidden semi-Markov models. Therefore, implementing three different models enables us to generate three predictions, each specializing on a different section of the dependency structure. Those predictions can then be combined to get a final prediction sequence, as explained in the next section.

2.2.4 The Hidden Semi-Markov Model and Computational Methods

Amino acid and DNA sequences have been successfully analyzed by hidden Markov models (HMM) as the character strings generated in the "left-to-right" direction. For a comprehensive introduction to HMMs, see [119].

In this work, we consider a hidden semi-Markov model (HSMM) also known as HMM with duration. Such type of model was earlier used in gene finding methods, such as Genie [89], GenScan [35] and GeneMark.hmm [33]. The HSMM technique was introduced for protein structure prediction by Schmidler *et al.* [134]. In a HSMM, a transition from a hidden state into itself cannot occur, while a hidden state can emit a

Table 13: Positional dependencies within loop segments for the models $\mathcal{M}1$, $\mathcal{M}2$, and $\mathcal{M}3$. Segments with L or less residues are considered. l is the segment length. $h_j^3 \in \{hydrophobic, neutral, hydrohilic\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$. h_j^5 is a five letter alphabet with groups defined as $\{P, G\}$, $\{E, K, R, Q\}$, $\{D, S, N, T, H, C\}$, $\{I, V, W, Y, F\}$, $\{A, L, M\}$.

loop		$\mathcal{M}1$	$\mathcal{M}2$	$\mathcal{M}3$
l=1	N1	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+1}^5	h_{i+1}^5, h_{i+2}^5
l=2	N1	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+1}^5	h_{i+1}^5, h_{i+2}^5
	C1	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+1}^5	h_{i+1}^5, h_{i+2}^5
l=3	N1	h_{i-1}^5, h_{i-2}^3	h_{i-1}^5, h_{i+1}^3	h_{i+1}^5, h_{i+2}^3
	N2	h_{i-1}^5, h_{i-2}^3	h_{i-1}^5, h_{i+1}^3	h_{i+1}^5, h_{i+2}^3
	C1	h_{i-1}^5, h_{i-2}^3	h_{i-1}^5, h_{i+1}^3	h_{i+1}^5, h_{i+2}^3

whole string of symbols rather than a single symbol. The hidden states of the model used in protein secondary structure prediction are the structural states {H, E, L} designating α-helix, β-strand, and loop segments, respectively. Here, state transitions occur with probabilities $P(T_j \mid T_{j-1})$, thus forming a first-order Markov chain. At each hidden state, an amino acid segment with uniform structure is generated according to the length distribution $P(S_j \mid S_{j-1}, T_j)$, and the likelihood $P(\mathbf{R}_{[S_{j-1}+1:S_j]} | S_{j-1}, S_j, T_j)$, as shown in Figure 5.

2.2.4.1 MAP vs MPM Estimation

Having defined this HSMM, we can consider the protein secondary structure prediction problem as the problem of finding the sequence of hidden states with the highest *a posteriori* probability given the amino acid sequence (MAP estimation). One efficient algorithm to solve this optimization problem is well known. Given an amino acid sequence $\mathbf{R}$, the vector $(\mathbf{S}, \mathbf{T})^* = \arg\max P(\mathbf{S}, \mathbf{T} \mid \mathbf{R})$ can be found using the Viterbi algorithm. Here lies a subtle difference between the result that can be delivered by the Viterbi algorithm and the result needed in the traditional statement of the protein secondary structure prediction problem. The Viterbi path does not

Table 14: Positional dependencies within structural segments for the first five amino acids of the protein (N-terminal). Models $\mathcal{M}1$ and $\mathcal{M}2$ contain $i-$ dependencies. $\varnothing$ denotes the empty set. $h_j^3 \in \{hydrophobic, neutral, hydrohilic\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$. h_j^5 is a five letter alphabet with groups defined as $\{P, G\}$, $\{E, K, R, Q\}$, $\{D, S, N, T, H, C\}$, $\{I, V, W, Y, F\}$, $\{A, L, M\}$.

		$\mathcal{M}1$	$\mathcal{M}2$
H	Int	h^3_{i-2}, h^3_{i-3}, h^3_{i-4}	h^3_{i+2}, h^3_{i+4}
	N1	$\varnothing$	h^3_{i+2}
	N2	h^3_{i-1}	h^3_{i+2}
	N3	h^3_{i-2}	h^3_{i+2}
	N4	h^3_{i-2}	h^3_{i+2}
	C1	h^3_{i-1}	h^3_{i+1}
	C2	h^3_{i-2}	h^3_{i+1}
	C3	h^3_{i-2}	h^3_{i+2}
	C4	h^3_{i-2}	h^3_{i+2}
E	Int	h^3_{i-1}, h^3_{i-2}	h^3_{i-1}, h^3_{i-2}
	N1	$\varnothing$	h^3_{i+1}
	N2	h^3_{i-1}	h^3_{i+2}
	C1	h^3_{i-1}	h^3_{i+2}
	C2	h^3_{i-1}	h^3_{i+2}
L	Int	h^3_{i-1}, h^3_{i-2}, h^3_{i-3}	h^3_{i+1}, h^3_{i+2}
	N1	$\varnothing$	h^3_{i+1}
	N2	h^3_{i-1}	h^3_{i+1}
	N3	h^3_{i-1}	h^3_{i+1}
	N4	h^3_{i-1}	h^3_{i+1}
	C1	h^3_{i-1}	h^3_{i+1}
	C2	h^3_{i-1}	h^3_{i+3}
	C3	h^3_{i-1}	h^3_{i+1}
	C4	h^3_{i-1}	h^3_{i+1}

Table 15: Positional dependencies within structural segments for the last five amino acids of the protein (C-terminal). Models $\mathcal{M}1$ and $\mathcal{M}2$ contain $i-$ dependencies. $\varnothing$ denotes the empty set. $h_j^3 \in \{hydrophobic, neutral, hydrohilic\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$. h_j^5 is a five letter alphabet with groups defined as $\{P, G\}$, $\{E, K, R, Q\}$, $\{D, S, N, T, H, C\}$, $\{I, V, W, Y, F\}$, $\{A, L, M\}$.

		$\mathcal{M}2$	$\mathcal{M}3$
H	Int	h^3_{i-2}, h^3_{i-4}	h^3_{i+2}, h^3_{i+4}
	N1	h^3_{i-2}	$\varnothing$
	N2	h^3_{i-2}	h^3_{i+1}
	N3	h^3_{i-3}	h^3_{i+2}
	N4	h^3_{i-2}	h^3_{i+2}
	C1	h^3_{i-2}	$\varnothing$
	C2	h^3_{i-2}	h^3_{i+1}
	C3	h^3_{i-2}	h^3_{i+2}
	C4	h^3_{i-2}	h^3_{i+2}
E	Int	h^3_{i-1}, h^3_{i-2}	h^3_{i+1}, h^3_{i+2}
	N1	h^3_{i+1}	$\varnothing$
	N2	h^3_{i+2}	h^3_{i+1}
	C1	h^3_{i+2}	$\varnothing$
	C2	h^3_{i+2}	h^3_{i+1}
L	Int	h^3_{i+1}, h^3_{i+2}	h^3_{i+1}, h^3_{i+2}, h^3_{i+3}
	N1	h^3_{i+1}	$\varnothing$
	N2	h^3_{i+1}	h^3_{i+1}
	N3	h^3_{i+1}	h^3_{i+1}
	N4	h^3_{i+1}	h^3_{i+1}
	C1	h^3_{i+1}	$\varnothing$
	C2	h^3_{i+1}	h^3_{i+1}
	C3	h^3_{i+1}	h^3_{i+1}
	C4	h^3_{i+1}	h^3_{i+1}

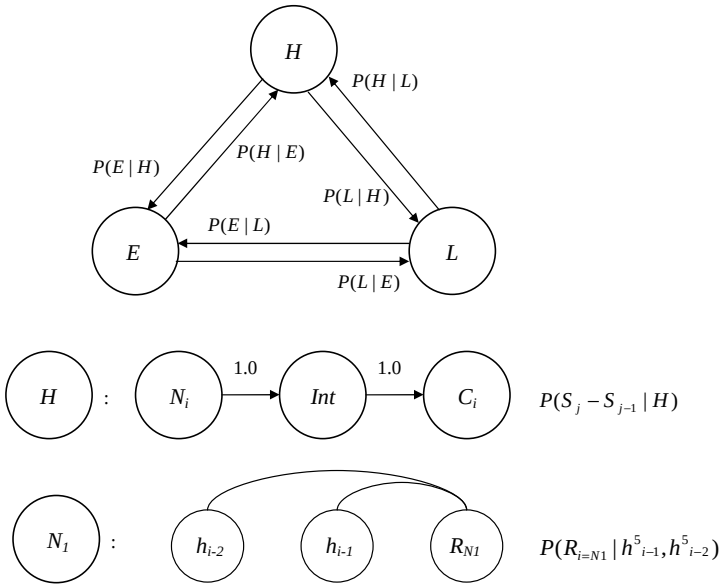


Figure 5: HSMM architecture. Transitions between secondary structure states are modeled as first order Markovian (top figure). Each state contains separate models for terminal and internal positions (middle figure). Position specific models have characteristic dependency structures with conditional independence of the amino acids (e.g. bottom figure shows dependency diagram for the N_1 residue of a structural segment under the model M_1).

directly optimize the three-state-per residue accuracy, (Q_3), which is defined as

$$Q_3 = \frac{\#\text{ correctly predicted structural states}}{\#\text{ observed symbols (amino acids)}}. \tag{8}$$

Also, the Viterbi algorithm might generate many different segmentations, which might have significant probability mass but are not optimal [134]. As an alternative to the Viterbi algorithm, we can determine the sequence of structural states that are most likely to occur in each position. This approach is also known as the marginal posterior mode (MPM) estimation, which utilizes forward and backward algorithms to compute the optimum prediction. Although this prediction might not be a perfectly valid state sequence (*i.e.*, it might not be realized given the parameters of HSMM), the prediction measure defined as the marginal posterior probability distribution correlates very strongly with the three-state-per-residue accuracy [134]. The performance of the Viterbi and forward-backward algorithms are compared in Schmidler *et al.* [134].

2.2.4.2 MPM Estimation and Forward Backward Algorithm

In this work, we used the marginal posterior mode (MPM) estimation approach to compute the optimum secondary structure segmentation. We define define forward and backward variables as

$$\begin{aligned}
\alpha_\theta(1,t) &= P_\theta(R_{[1]} \mid T=t, S=1) P_\theta(S=1 \mid T=t) P_\theta(T=t) \qquad (9)\\
\alpha_\theta(j,t) &= P_\theta(R_{[1:j]}, S=j, T=t) \\
&= \sum_{v=1}^{j-1} \sum_{l \in SS} \alpha_\theta(v,l) P_\theta(R_{[v+1:j]} \mid S_{prev}=v, S=j, T=t) \\
&\times \; P_\theta(S=j \mid T=t, S_{prev}=v) P_\theta(T=t \mid T_{prev}=l) \\
j &= 2, ..., n
\end{aligned}$$

$$\begin{aligned}
\beta_\theta(n,t) &= 1 \qquad (10)\\
\beta_\theta(j,t) &= P_\theta(R_{[j+1:n]} \mid S=j, T=t) \\
&= \sum_{v=j+1}^{n} \sum_{l \in SS} \beta_\theta(v,l) P_\theta(R_{[j+1:v]} \mid S_{prev}=v, S=j, T_{next}=l) \\
&\times \; P_\theta(S_{next}=v \mid S=j, T_{next}=l) P_\theta(T_{next}=l \mid T=t) \\
j &= n-1, ..., 1
\end{aligned}$$

In the above formulations, the forward variable $\alpha_\theta(j,t)$ is the joint probability of observing the amino acid sequence up to position j and a secondary structure segment that ends at position j with type t. Here, θ represents the statistical dependency model. Similarly, the backward variable $\beta_\theta(j,t)$ defines the conditional probability of observing the amino acid sequence in positions $j+1$ to n and a secondary structure segment that ends at position j with type t.

Having defined the forward and backward parameters, the *a posteriori* probability for a hidden state in position i to be either an α-helix, β-strand or loop is computed via all possible segmentations that include position i as formulated in Eq. (11). The

hidden state at position i is inferred as the state with maximum *a posteriori* probability. Finally, the whole predicted sequence of hidden states is defined by Eq. (12).

$$\begin{aligned} P_\theta(T_{R_i} \mid R) &= \sum_{j=1}^{i-1} \sum_{k=i}^{n} \sum_{l \in SS} \alpha_\theta(j,l)\beta_\theta(k,t)P_\theta(T = t \mid T_{prev} = l) \\ &\times \; P_\theta(S = k \mid S_{prev} = j, T = t) \\ &\times \; P_\theta(R_{[j+1:k]} \mid S_{prev} = j, S = k, T = t)/P_\theta(R) \end{aligned} \tag{11}$$

$$(S,T)^* = \arg\max_{(S,T)} \{P_\theta(T_{R_i} \mid R)\}_{i=1}^{n} \tag{12}$$

The computational complexity of this algorithm is $O(n^3)$. If the maximum allowed size of a segment is chosen as D, the first summation in Eq. (9) starts at $(j - D)$, and the first summation in Eq. (10) ends at $(j + D)$ reducing the computational cost to $O(nD^2)$.

Scaling Forward and backward variables are computed by multiplying probabilities, which are less than one, and as the sequence gets longer, these variables approximate to zero after a certain position. Hence, it is necessary to introduce a scaling procedure to prevent numerical underflow. The scaling for a "classic" HMM is described in [119]. This procedure can easily be generalized for an HSMM, where the scaling coefficients are introduced at every D positions. In that case the forward equations take the

following form:

$$
\begin{aligned}
\hat{\alpha}_\theta(1,t) &= P_\theta(R_{[1]} \mid T=t, S=1) P_\theta(S=1 \mid T=t) P_\theta(T=t) \qquad (13)\\
\bar{\alpha}_\theta(j,t) &= c_\alpha(L_1+1,t) \sum_{v=j-D}^{L_1 \geq v} \sum_{l \in SS} \hat{\alpha}_\theta(v,l) P_\theta(R_{[v+1:j]} \mid S_{prev}=v, S=j, T=t)\\
&\times \; P_\theta(S=j \mid T=t, S_{prev}=v) P_\theta(T=t \mid T_{prev}=l)\\
&+ \sum_{v=L_1+1}^{j-1} \sum_{l \in SS} \hat{\alpha}_\theta(v,l) P_\theta(R_{[v+1:j]} \mid S_{prev}=v, S=j, T=t)\\
&\times \; P_\theta(S=j \mid T=t, S_{prev}=v) P_\theta(T=t \mid T_{prev}=l)\\
L_1 &= D \times \lfloor \frac{(j-2)}{D}\\
\hat{\alpha}_\theta(j,t) &= c_\alpha(j,t) \bar{\alpha}_\theta(j,t)\\
c_\alpha(j,t) &= \begin{cases} \frac{1}{\bar{\alpha}_\theta(j,H)+\bar{\alpha}_\theta(j,E)+\bar{\alpha}_\theta(j,L)} & \text{if } j = aD+1 \\ 1 & \text{o/w} \end{cases}\\
j &= 2, ..., n
\end{aligned}
$$

In the above formulations, $\hat{\alpha}_\theta(j,t)$ is the scaled forward variable, $c_\alpha(j,t)$ is the scaling coefficient, $\lfloor$ is the operator that rounds to the smaller integer, and a is a nonnegative integer. One can prove that:

$$\hat{\alpha}_\theta(j,t) = \left(\prod_{k=0}^{L_2} c_\alpha(kD+1,t) \right) \alpha_\theta(j,t), \qquad (14)$$

where $L_2 = \lfloor \frac{(j-1)}{D}$ and $\alpha_\theta(j,t)$ is the original unscaled forward variable. Similar to the forward variable, the backward variable can be reexpressed as

$$
\begin{aligned}
\hat{\beta}_\theta(n,t) &= 1 \qquad (15)\\
\tilde{\beta}_\theta(j,t) &= \sum_{v=j+1}^{n-L_3-1\geq v} \sum_{l\in SS} \hat{\beta}_\theta(v,l) P_\theta(R_{[j+1:v]} \mid S_{prev}=v, S=j, T_{next}=l)\\
&\times\ P_\theta(S_{next}=v \mid S=j, T_{next}=l) P_\theta(T_{next}=l \mid T=t)\\
&+\ c_\beta(n-L_3-1,t) \sum_{v=n-L_3}^{j+D\geq v} \sum_{l\in SS} \hat{\beta}_\theta(v,l) P_\theta(R_{[j+1:v]} \mid S_{prev}=v, S=j, T_{next}=l)\\
&\times\ P_\theta(S_{next}=v \mid S=j, T_{next}=l) P_\theta(T_{next}=l \mid T=t)\\
L_3 &= D \times \lfloor \frac{(n-j-2)}{D}\\
\hat{\beta}_\theta(j,t) &= c_\beta(j,t)\tilde{\beta}_\theta(j,t)\\
c_\beta(j,t) &= \begin{cases} \frac{1}{\tilde{\beta}_\theta(j,H)+\tilde{\beta}_\theta(j,E)+\tilde{\beta}_\theta(j,L)} & \text{if } n-j = aD+1 \text{ and } (n-j) > D \\ 1 & \text{o/w} \end{cases}\\
j &= n-1, ..., 1
\end{aligned}
$$

Here, $\hat{\beta}_\theta(j,t)$ is the scaled backward variable, and $c_\beta(j,t)$ is the scaling coefficient. As in the forward variable, we can prove that:

$$\hat{\beta}_\theta(j,t) = \left(\prod_{k=0}^{L_4} c_\beta(n-kD-1,t) \right) \beta_\theta(j,t), \qquad (16)$$

where $L_4 = \lfloor \frac{(n-j-1)}{D}$ and $\beta_\theta(j,t)$ is the original unscaled backward variable.

Once the scaled versions of the forward and backward variables are obtained, the marginal posterior probability can be computed as

$$
\begin{aligned}
Q(T_{R_i}, R) &= \sum_{j=1}^{i-1} \sum_{k=i}^{n} \sum_{l\in SS} \hat{\alpha}_\theta(j,l) \hat{\beta}_\theta(k,t) P_\theta(T=t \mid T_{prev}=l) \qquad (17)\\
&\times\ P_\theta(S=k \mid S_{prev}=j, T=t)\\
&\times\ P_\theta(R_{[j+1:k]} \mid S_{prev}=j, S=k, T=t)\\
P_\theta(T_{R_i} \mid R) &= Q(T_{R_i}, R) / \sum_{T_{R_i}} Q(T_{R_i}, R)
\end{aligned}
$$

,

where the forward and backward variables are replaced with their scaled versions and $Q(T_{R_i}, R)$ is the scaled version of $P(T_{R_i}, R)$, *i.e.*, the numerator in Eq. 11. When the posterior distribution is available the secondary structure prediction can be computed easily as in Eq. 12.

2.2.4.3 Combined Model

This completes the derivation of the algorithm for a single model. Since we are utilizing three dependency models, *i.e.*, $\theta = \mathcal{M}1, \mathcal{M}2, \mathcal{M}3$, it becomes necessary to combine the outputs of those models to get a single prediction. In our simulations, we implemented averaging and maximum operators to perform this task and observed that the averaging function gives the best performance. The final prediction sequence is then computed as

$$\begin{aligned} P^C(T_{R_i} \mid R) &= (P^{\mathcal{M}1}(T_{R_i} \mid R) + P^{\mathcal{M}2}(T_{R_i} \mid R) + P^{\mathcal{M}3}(T_{R_i} \mid R))/3 \\ (S,T)^* &= \arg\max_{(S,T)} \left\{ P^C(T_{R_i} \mid R) \right\}_{i=1}^{n} \end{aligned} \tag{18}$$

2.3 Model Training

Having derived the HSMM, we need to estimate the model parameters so that we can compute predictions using the algorithms described in Section 2.2.4. The model parameters are mainly the transition, amino acid observation (emission) and the length distributions explained in Sections 2.2.4 and 2.2. The parameters of the HSMM can be estimated by various techniques. First of all, we assume that we have a set of example sequences known as training sequences, which are of the type that we want our model to fit well. When the paths (or state sequences) are known for all the examples (or training sequences), then we can perform maximum likelihood parameter estimation, which is a supervised learning approach.

Let our model be $\mathcal{M}$ and let the set of training sequences be $D = \{x^1, ..., x^n\}$. We

assume that the elements of D are independent and thus define the joint probability of all the sequences given a particular assignment of parameters as the product of the probabilities of the individual sequences. This is formulated as

$$P(x^1, ..., x^n \mid \theta, \mathcal{M}) = \prod_{i=1}^{n} P(x^i \mid \theta, \mathcal{M}), \tag{19}$$

where θ represents the entire set of values of the parameters in the model. Then, the maximum likelihood estimator is formulated as

$$\theta^{ML} = \arg\max_{\theta} P(D \mid \theta, \mathcal{M}). \tag{20}$$

When all the state sequences of the training data is available, then the ML estimator can be easily computed using the frequency of occurrence counts. Let $P(T_{next} \mid T_{pre})$ be the probability of making a transition from state T_{pre} to T_{next}, with T_{pre} and $T_{next} \in \{H, E, L\}$. Then the maximum likelihood estimator for $P(T_{next} \mid T_{pre})$ becomes:

$$P^{ML}(T_{next} \mid T_{pre}) = \frac{\#\text{ transactions from state } T_{pre} \text{ to } T_{next}}{\#\text{ visits to } T_{pre} \text{ followed by another state}}. \tag{21}$$

Since there are three possible secondary structure states and self transitions are not allowed in an HSMM then we have a total of $3 * 3 - 3 = 6$ possible transition parameters. Similar to the transition probability distribution, we can compute the amino acid observation and length distributions. For instance, $P^H_{N_3}(R_i \mid h^3_{i-2}, h^3_{i-3}, h^3_{i+2})$ is computed as the total number of R_i occurrence in position N3 of an α-helix segment, with the hydrophobicity of the amino acids at positions $i-2, i-3$ and $i+2$ are equal to $h^3_{i-2}, h^3_{i-3}, h^3_{i+2}$, respectively, divided by the total number of $h^3_{i-2}, h^3_{i-3}, h^3_{i+2}$ occurrence. Here, h^3_{i-2} is the hydrophobicity class of the amino acid at position $i-2$ (see the hydrophobicity definitions in Table 10).

Since the true secondary structures are available in the PDB database, we used the maximum-likelihood estimation procedure to derive the HSMM parameters where we count the observed frequencies for the desired quantities, and apply a proper normalization factor to compute the probability values.

2.3.1 Training Set Reduction

To improve the estimation of the HSMM parameters, we implemented a training set reduction approach. Once we obtain a prediction for the input sequence, we compute the similarity scores between the input sequence and training sequences. Then, from the training set, we remove sequences that are not similar to the input sequence. The dataset reduction step is followed by the re-estimation of the HSMM parameters and the prediction of the secondary structure as shown in Figure 6. This approach allows us to build less contaminated models and obtain more accurate predictions.

There can be various techniques to remove sequences that are dissimilar to the input sequence. In this section, we compared three dataset reduction methods to refine the parameters of an HSMM: (1) composition based reduction; (2) alignment based reduction; and (3) reduction using Chou-Fasman parameters. In each method, the dataset reduction is based on a similarity (or a distance) measure. We considered two decision functions to classify proteins as similar or dissimilar. The first function selects the top 80% of the proteins in the original training set that are similar to the input protein and the second function selects proteins according to a threshold.

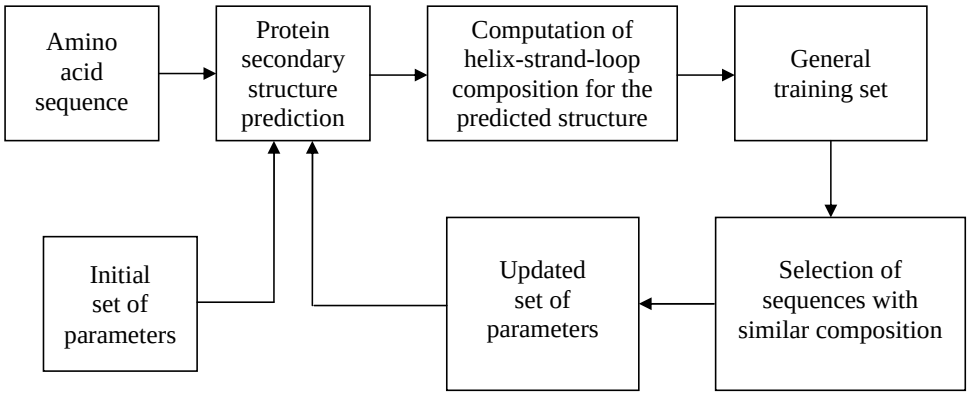


Figure 6: Training set reduction procedure. Initial set of model parameters is precomputed from the general training set.

2.3.1.1 Composition Based Reduction

In this method, the distance between the predicted secondary structure and the secondary structure segmentation of a training data is computed as follows:

$$D = \max(|H_p - H_t|, |E_p - E_t|, |L_p - L_t|), \tag{22}$$

where H_p, E_p, and L_p denote the composition of the α-helices, β-strands, and loops in the predicted secondary structure, respectively. Similarly H_t, E_t, and L_t represent the composition of the α-helices, β-strands, and loops in the training data. Here, the composition is defined as the ratio of the number of secondary structure symbols in a given category to the length of the protein. For instance, H_p is equal to the number of α-helix predictions divided by the total number of amino acids in the input protein. After sorting the proteins in the training set, we considered two decision functions to construct the reduced set: (1) selection of the first 80% of the proteins with the lowest D values; (2) selection of the proteins that satisfy $D < 0.35$ [3].

2.3.1.2 Alignment Based Reduction

In this reduction scheme, first, pairwise alignments of the input protein to training set proteins are computed. Then, proteins with low alignment scores are excluded from the training set. As in the composition based method, two approaches are considered to obtain the reduced dataset: (1) selection of the first 80% of the proteins with the highest alignment scores; (2) selection of the proteins with alignment scores above a threshold. Here, the threshold is computed by finding the alignment score that corresponds to the threshold used in the composition based reduction method. In the following sections, we will give more details on the pairwise alignment implementation.

Alignment Scenarios We considered the following cases:

[3] The threshold is found empirically [23].

- Alignment of secondary structures (SS)
- Alignment of amino acid sequences (AA)
- Joint alignment of amino acid sequences and secondary structures (AA+SS)

In the first case, the aligned symbols are the secondary structure states, which take one of the three values: H, E, or L. In the second case, the symbols are the amino acids and finally, in the third case, the aligned symbols are amino acid and secondary structure pairs.

Score Function The score of an alignment is computed by summing the scores of the aligned symbols (matches and mismatches) as well as the gapped regions. This is formulated as follows:

$$S = \sum_{k=1}^{r} (\alpha M_{aa}(a_k, b_k) + \beta M_{ss}(c_k, d_k)) + G, \tag{23}$$

where S is the alignment score, r is the total number of match/mismatch pairs, G is the total score of the gapped regions, a_k, b_k represent the k^{th} amino acid pair of the aligned proteins (the input and the training set protein, respectively), c_k, d_k denote the k^{th} secondary structure pair of the aligned proteins, $M_{aa}(.)$ is the amino acid similarity matrix, $M_{ss}(.)$ is the secondary structure similarity matrix, and finally, the parameters α, and β determine the weighted importance of the amino acid and secondary structure similarity scores, respectively. To compute possible alignment variations described in the previous section, α and β take the following values: (1) $\alpha = 0, \beta = 1$ to align secondary structures; (2) $\alpha = 1, \beta = 0$ to align amino acid sequences; (3) $\alpha = 1, \beta = 1$ to align amino acid and secondary structures in jointly.

Similarity Matrices We used the BLOSUM30 table [68] as the amino acid similarity matrix and the Secondary Structure Similarity Matrix (SSSM) [144] shown in Table 16.

Table 16: Secondary structure similarity matrix, which is used to score the similarity of two secondary structure symbols.

M_{ss}	H	E	L
H	2	-15	-4
E	-15	4	-4
L	-4	-4	2

Gap Scoring When a symbol in one sequence does not have any counterpart (or match) in the other sequence, then that symbol is aligned to a gap symbol '-'. Allowing gap regions in an alignment enables us to represent the similarity between the aligned sequences in a biologically meaningful manner. In the state-of-the-art gap scoring, opening a gap is penalized more than extending it. For example, in the "affine gap scoring", which is one of the most widely used gap scoring techniques, starting a gap is scored by the parameter g_o, and extending a gap region is scored by g_e. In that case, the total gap score in (23) is computed as

$$G = N_o g_o + N_e g_e, \tag{24}$$

where N_o is the total number of gap openings, and N_e is the total number of gap extensions. In this work, we set the parameters g_o, and g_e to -12, and -2, respectively.

Optimum Alignment Given a scoring function, the computation of the optimum (best scoring) alignment can be found using a dynamic programming approach. In this section, we used the Smith-Waterman algorithm to compute the local alignment between a pair of proteins. Further details on the alignment algorithms and dynamic programming can be found in Durbin *et al.* [53].

Score Normalization After computing the raw score of an alignment, it is useful to normalize it to a statistically meaningful range. In this section, we normalized the alignment score by the average length of the aligned proteins. In that case, the normalized score is computed as $2\frac{rawscore}{l_1+l_2}$, where l_1, and l_2 are the lengths of the

aligned proteins. This type of normalization is shown to be effective in fold recognition by Aydin *et al.* [25].

2.3.1.3 Reduction using Chou-Fasman Parameters

In this approach, the training set reduction is based on the Chou-Fasman distance measure, which is defined as

$$D_{cf} = \sum_{k \in H,E,L} \left\{ \frac{1}{l_p} \sum_{j=1}^{l_p} f_k(q(j)) - \frac{1}{l_t} \sum_{j=1}^{l_t} f_k(h(j)) \right\}, \tag{25}$$

where l_p is the length of the input protein, l_t is the length of the training set protein, $q(j)$ is the j^{th} amino acid of the input protein, $h(j)$ is the j^{th} amino acid of the training set protein, and $f_k(z)$ is the Chou-Fasman coefficient that reflects the propensity of the amino acid of type z to be in the secondary structure state k. These coefficients can be computed as in [42]. In this formulation, the secondary structure information of the proteins is not used and each amino acid is allowed to take three possible secondary structure states. In a slightly modified version of this method, we define the Chou-Fasman distance using the secondary structure information as follows:

$$D_{cf,2} = \left\{ \frac{1}{l_p} \sum_{j=1}^{l_p} f_{k(q(j))}(q(j)) - \frac{1}{l_t} \sum_{j=1}^{l_t} f_{k(h(j))}(h(j)) \right\}, \tag{26}$$

where $k(q(j))$ is the predicted secondary structure state for the j^{th} amino acid of the input protein, and $k(h(j))$ is the secondary structure state for the j^{th} amino acid of the training set protein. In Chou-Fasman based reduction, we computed the reduced dataset by selecting the first 80% of the proteins with the lowest Chou-Fasman distances and did not perform threshold based reduction.

2.4 Iterative Protein Secondary Structure Parse (IPSSP) Algorithm

We developed the IPSSP algorithm, which implements the methods described in the previous sections. IPSSP utilizes three HSMMs and the composition based reduction

scheme because in our simulations, the composition based reduction gave the most accurate results (see Section 2.5). The steps of the algorithm can be summarized as follows:

Algorithm 1: IPSSP Algorithm

Input: Amino acid sequence **R** , Training set D
Output: Secondary Structure Prediction **SS**

```
for each HSMM do
    Compute the posterior probability distribution using the posterior decoding
    algorithm (i.e., the forward-backward algorithm);
    Compute a secondary structure prediction by selecting the most likely state
    for each amino acid;
    Reduce the original training set D using the composition based reduction;
    Train the HSMM with the reduced dataset;
    Compute the posterior probability distribution;
Take the average of the three posterior probability distributions;
Compute the final prediction SS;
```

2.5 Simulation Results

2.5.1 Experimental Settings

2.5.1.1 Datasets

EVA SET The EVA set is derived from the PDB database [15]. The proteins in the EVA set are selected to satisfy the condition that percentage of identity between any pair of sequences should not exceed the length dependent threshold S (for instance, for sequences longer than 450 amino acids, $S = 19.5$) [124]. The EVA set contains 3324 "sequence-unique" proteins dated as 2004_05_09 and can be downloaded from the EVA server ftp site [10]. In our simulations, we removed sequences shorter than 30 amino acids and arrived to a set of 2720 proteins.

PDB_SELECT Dataset The PDB_SELECT dataset contains a representative set of 2482 amino acid sequences dated as 2005 [13]. The procedure used to generate the PDB_SELECT list is described earlier [69]. In this set, the percentage of identity between any pair of sequences is less than 25%.

CASP6 Targets CASP6 targets were downloaded from [3], and the PDB definitions were used for the amino acid sequences and secondary structure assignments.

Dataset used by the PSIPRED method PSIPRED training data was downloaded from [17].

CB513 Set The set (CB513) of 513 sequences with essentially no similarity to each other (non-homologous) is introduced in [49] and can be downloaded from [4].

2.5.1.2 Accuracy Measures

Sensitivity We use the three-state-per-residue accuracy (Q_3), defined in Eq. (27) as the overall sensitivity measure:

$$Q_3(\%) = \frac{N_c}{N} \times 100, \tag{27}$$

where N_c is the total number of residues with correctly predicted secondary structure and N is the total number of amino acids observed in the test data. The same measure can also be used for each type of secondary structure, Q_α, Q_β, and Q_L as expressed in Eq. (28):

$$Q_i(\%) = \frac{N_c^i}{N^i} \times 100, \tag{28}$$

where N_c^i is the total number of residues with correctly predicted secondary structure of type i and N^i is the total number of amino acids observed in the conformation of type i. The sensitivity measure can also be formulated as TP/(TP+FN), where TP is the true positives and FN is the false negatives.

Positive Predictive Value The positive predictive value (ppv) measure PPV_i is defined for individual types of secondary structure as follows:

$$PPV_i(\%) = \frac{N_c^i}{N_p^i}, \tag{29}$$

where where N_c^i is the total number of residues with correctly predicted secondary structure of type i and N_p^i is the total number of amino acids predicted to be in conformation of type i. Note that we do not consider the overall ppv measure PPV_3, since its numeric value is the same as Q_3. The positive predictive value measure can also be defined as TP/(TP+FP), where TP is the true positives and FP is the false positives.

Matthew's Correlation Coefficient The Matthew's correlation coefficient [95] is a single parameter characterizing the extent of a match between the observed and the predicted secondary structure. Matthew's correlation is defined for each type of secondary structure as follows:

$$MCC = \frac{TP * TN - FP * FN}{[(TN + FN)(TN + FP)(TP + FN)(TP + FP)]^{1/2}} \tag{30}$$

For instance, for an α-helix, TP (true positives) is the number of α-helix residues that are correctly predicted. TN (true negatives) is the number of residues observed in β-strands and loops that are not predicted as α-helix. FP (false positives) is the number of residues incorrectly predicted in α-helix conformation, and finally FN (false negatives) is the number of residues observed in α-helices but predicted to be either in β-strands or loops.

Segment OVerlap Score (SOV) The Segment OVerlap score (SOV) is based on the average overlap between the observed and the predicted segments instead of the average per-residue accuracy [129], [151]. The SOV measures provide more elaborate scoring in which the predictions that have high per-residue accuracy but deviate from experimental segment length distributions are assigned lower scores. For instance, the definition of the SOV measure for α-helices is as follows:

$$SOV_\alpha = \frac{1}{N_\alpha} \sum_{S_\alpha} \frac{minOV(s_1, s_2) + \delta(s_1, s_2)}{maxOV(s_1, s_2)}, \tag{31}$$

where s_1 and s_2 are the observed and predicted secondary structure segments in the α-helix state; S_α is the number of all pairs of segments s_1 and s_2 such that s_1 and s_2 have at least one residue in α-helix state in common, $minOV(s_1, s_2)$ is the length of the actual overlap of s_1 and s_2, $maxOV(s_1, s_2)$ is the length of the total extent for which either of the segments s_1 or s_2 has a residue in the α-helix state, and N_α is the total number of amino acid residues observed in α-helix conformation. The definition of $\delta(s1, s2)$ is as follows [151]:

$$\delta(s_1, s_2) = min \left\{ \begin{array}{c} maxOV(s_1, s_2) - minOV(s_1, s_2) \\ minOV(s_1, s_2) \\ int(0.5 \times len(s_1)) \\ int(0.5 \times len(s_2)), \end{array} \right\} \tag{32}$$

where $len(s_1)$ is the number of amino acid residues in s_1. The segment overlap measure for all three states, $SOV_3(\%)$, is similar to the $Q_3(\%)$ sensitivity measure:

$$SOV_3(\%) = \frac{1}{N} \left(\sum_{i \in H,E,L} \sum_{S(i)} \left[\frac{minOV(s_1, s_2) + \delta(s_1, s_2)}{maxOV(s_1, s_2)} \times len(s_1) \right] \right) \times 100, \tag{33}$$

where s_1 and s_2 are the observed and predicted secondary structure segments in state i and N is the total number of amino acids in all proteins that are evaluated.

2.5.1.3 Performance Evaluation and Cross Validation

In machine learning, first, the model parameters are derived from a training set. Then, the prediction accuracy is assessed on test samples with known state definitions.

In a cross validation experiment, a set is typically partitioned into k subsets. Of the k subsets, a single subset is retained as the validation data for testing the model, and the remaining $k - 1$ subsets are used as training data. The cross-validation process is then repeated k times, with each of the k subsets used exactly once as the validation data. The k results from the folds can then be averaged (or otherwise combined) to produce a single estimation. When k equals to the number of examples in the dataset, this is called leave-one-out cross validation (or jacknife procedure).

In our simulations, we used two types of performance evaluation practices: (1) Selecting separate training and test sets and evaluating the accuracy (*e.g.* CASP6 as the test set and PDB_SELECT as the training set); (2) Leave-one-out cross validation on a single dataset. In latter case, we first select a protein as the test example and remove it from the dataset. The remaining proteins form the training set and are used to estimate the parameters of the hidden semi-Markov model (*i.e.*, transition, length and emission distributions). Since the true secondary structures are available, we can use the maximum-likelihood estimation procedure, in which the observed frequencies for the desired quantities are divided by a proper normalization factor to compute the probability values (see Section 2.3). After estimating the model parameters, we predict the secondary structure sequence of the test protein. Then, we include the test protein into the dataset, select another test protein and repeat the prediction until all the proteins in the dataset are evaluated. Finally, we compute the performance measures by taking the true secondary structures of the proteins as reference.

2.5.1.4 State Reduction and Length Adjustments

The secondary structure is assigned from the experimentally determined 3-D structure. The assignment of secondary structure is based on the detection of hydrogen bonds between the amino acids given a set of 3-D atomic coordinates. Among the best assignment algorithms, we can mention DSSP [81], STRIDE (Frishman and Argos, 1995) and DEFINE (Richards and Kundrot, 1988). In this thesis, we use DSSP since it has been the most widely used secondary structure definition. It has eight secondary structure states defined as: H(α-helix), G(3_{10}-helix), I(πhelix), E(β-strand), B(isolated β-bridge), T(turn), S(bend) and ' ' (rest). These eight states are often collapsed or reduced into three standard states (helices (H), strands (E), and loops (L) (or coils (C))) because the states have structural similarities and prediction in eight classes is technically more difficult.

The reduction from the eight-state alphabet to three-state representation is usually performed by a conversion rule. In this thesis, we considered the following three methods: (i) H, G, and I to H; E, B to E; all other states to L, (ii) H, G to H; E, B to E; all other states to L, (iii) H to H; E to E; all other states to L. The first rule is also known as the 'EHL' mapping [101, 126], the second rule is the one used in PSIPRED [80] and earlier outlined by Rost and Sander [128], while the third rule is the common 'CK' mapping, which is the one used in BSPSS and other methods [38, 59, 49].

After applying either of the three conversion rules, we considered making further adjustments. We used the adjustments proposed by Frishman and Argos [58] that lead to a secondary structure sequence with the minimum β-strand length of three and the minimum α-helix length of five.

2.5.1.5 Single-sequence vs. sequence-unique condition

We would like to emphasize that, we use the term single-sequence prediction in its strict meaning, *i.e.*, the prediction method does not exploit information about any protein sequence similar to the sequence in question as for a true single-sequence such information does not exist. The "single-sequence" concept should be distinguished from the concept of the "sequence-unique" category. The "sequence-unique" condition requires absence of significant similarity between the proteins in the test and in the training set. However, this condition leaves an opportunity to use the sequence profile information that typically improves the prediction accuracy by several percentage points in comparison with the single-sequence condition, in which such profiles are not available. Indeed, methods such as APSSP2 [120] and SVMpsi [84] achieved values around 78% in the "sequence-unique" category of CASP [6] and CAFASP [5] experiments. Similarly, the SSPAL method [132] was cited [134] to have 71% accuracy

in terms of $Q_3(\%)$ again in the "sequence-unique" category. Single-sequence condition, as defined, is more stringent. This condition is common for "orphan" proteins, which have no detectable homologs. Improvement of structural prediction under the single-sequence condition should contribute to the improvement of function prediction for orphan proteins, which are not easy targets for functional characterization.

2.5.2 Comparison with the State-of-the-Art

2.5.2.1 BSPSS vs. IPSSP

We first compared the performances of BSPSS [134] and IPSSP in terms of the following accuracy measures: the Sensitivity, positive predictive value, Matthew's correlation coefficient, and segment overlap score (see Section 2.5.1.2. In our computations, we used the EVA set of "sequence-unique" proteins derived from the PDB database. For the IPSSP method, we applied composition based reduction and used a threshold of 0.35 in the dataset reduction step (see Section 2.3.1.1). For the maximum allowed segment length, we chose a threshold of $D = 50$, which is sufficiently large to cover almost all observed uniform secondary structure segments (see Section 2.2.1). Also, for longer segments, the maximum likelihood estimation for length distribution becomes less reliable because of the small sample size. The performances of IPSSP and BSPSS were evaluated by a leave-one-out cross validation experiment (jacknife procedure) on the reduced version of the EVA set (see Section 2.5.1). From the results shown in Table 17, there is a 1.9% increase in the overall three-state prediction accuracy in comparison with BSPSS, when the third conversion rule was used with the length adjustments. The prediction accuracy of the structural conformation of the residues

Table 17: Prediction sensitivity measures, $Q_.(\%)$, evaluated on the EVA set under the single-sequence condition.

Sensitivity	$Q_3(\%)$	$Q_\alpha(\%)$	$Q_\beta(\%)$	$Q_L(\%)$
BSPSS	68.400	63.203	36.737	**82.167**
IPSSP	**70.342**	**66.204**	**44.995**	81.358

situated close to structural segment borders (residues located in proximal positions) is measured by sensitivity values computed as overall Q_{3_sb} as well as structure-type specific Q_{α_sb}, Q_{β_sb}, Q_{L_sb}. We observed that the accuracy of IPSSP is better than BSPSS in proximal positions by 1.6% as shown in Table 18.

Table 18: Segment border sensitivity values, $Q_{.\_sb}(\%)$, evaluated on the EVA set under the single-sequence condition.

Sensitivity	$Q_{3\_sb}(\%)$	$Q_{\alpha\_sb}(\%)$	$Q_{\beta\_sb}(\%)$	$Q_{L\_sb}(\%)$
BSPSS	62.207	52.634	24.215	**81.903**
IPSSP	**63.959**	**55.941**	**32.439**	80.358

Next, we compared the positive predictive values of BSPSS and IPSSP. The results in Table 19 show that values of PPV_α and PPV_L are higher for IPSSP, while PPV_β value is higher for BSPSS.

Table 19: Positive predictive value measures, $PPV_.(\%)$, evaluated on the EVA set under the single-sequence condition.

Specificity	$PPV_\alpha(\%)$	$PPV_\beta(\%)$	$PPV_L(\%)$
BSPSS	68.636	**59.728**	69.832
IPSSP	**71.974**	59.686	**71.987**

The third accuracy measure is the Matthew's correlation coefficient (MCC). All the MCC values shown in Table 20 are higher for IPSSP.

Table 20: Matthew's correlation coefficient values, $C_.$, evaluated on the EVA set under the single-sequence condition.

MCC	C_α	C_β	C_L
BSPSS	0.5195	0.3849	0.4468
IPSSP	**0.5642**	**0.4316**	**0.4766**

The SOV scores of BSPSS and IPSSP are evaluated and compared on the EVA set. In terms of the Segment Overlap scores, IPSSP performs uniformly better than BSPSS as shown in Table 21.

Table 21: Segment overlap measures, $SOV\_(\%)$, for BSPSS and IPSSP evaluated on the EVA set under the single-sequence condition. To reduce eight states to three, the third conversion rule (CK mapping: H to H, E to E and all other states to L) is used.

SOV	$SOV_3(\%)$	$SOV_\alpha(\%)$	$SOV_\beta(\%)$	$SOV_L(\%)$
BSPSS	58.985	66.508	46.816	58.733
IPSSP	**63.616**	**69.965**	**54.176**	**63.141**

2.5.2.2 BSPSS, IPSSP and PSIPRED

We evaluated and compared the performances of BSPSS, IPSSP and PSIPRED_v2.0 on the set of 81 CASP6 targets (see Section 2.5.1.1) that are available in the PDB. This evaluation is at the "single-sequence condition" implying no additional evolutionary information is available. We used the software "PSIPRED_single", version 2.0, which uses a set of fixed weight matrices in the neural network and does not employ PSI-BLAST profiles. This program was downloaded from the PSIPRED server [16] with the available training data. We used the same training set to estimate the parameters of BSPSS and IPSSP (see Section 2.5.1). For the IPSSP method, we applied composition based reduction and used a threshold of 0.35 in the dataset reduction step (see Section 2.3.1.1). For the maximum allowed segment length, we chose a threshold of $D = 50$. From the results shown in Table 22, and Table 23, IPSSP is comparable to PSIPRED and is more accurate than BSPSS.

Table 22: Prediction sensitivity measures evaluated on the CASP6 targets.

Sensitivity	$Q_3(\%)$	$Q_\alpha(\%)$	$Q_\beta(\%)$	$Q_L(\%)$
BSPSS	66.541	75.177	41.743	72.696
IPSSP	**67.899**	74.984	46.087	**73.755**
PSIPRED	67.680	**76.066**	**52.032**	69.028

Table 23: Matthew's correlation coefficients evaluated on the CASP6 targets.

MCC	$C_\alpha(\%)$	$C_\beta(\%)$	$C_L(\%)$
BSPSS	0.5403	0.4354	0.4457
IPSSP	**0.5657**	0.4486	**0.4696**
PSIPRED	0.5465	**0.4801**	0.4646

2.5.2.3 IPSSP, GORIV, SOPM and SIMPA

In this set of simulations, we compared the performances of IPSSP and three state-of-the-art methods for single-sequence prediction: GORIV [60], SOPM [62], and SIMPA [91]. The evaluation of GORIV, SOPM, and SIMPA is available in Cuff and Barton [49], where the three-state-per-residue accuracies of the methods are reported as SIMPA: 67.6%, GORIV: 64.6%, and SOPM: 64.7% evaluated on the CB396 set, which is a subset of the CB513 set (see Section 2.5.1.1). The Cuff and Barton also evaluated the performance of the SOPM method on the CB513 set and obtained 65.7% as the $Q_3(\%)$ measure (a 1.0% improvement). However, a similar improvement was not observed for the SIMPA method (only 0.3%) when the results of the RS126 and CB396 sets are compared.

To evaluate the IPSSP method, we performed a leave-one-out cross validation on a smaller version of the CB513 set, which contains 494 proteins. This set is obtained by removing proteins with segment lengths longer than $D = 40$. To reduce eight secondary structure states to three, we used the CK mapping without length adjustments, which is the same mapping as in the Cuff and Barton evaluation [49]. Since the CB513 dataset contains fewer number of sequences than the EVA set and since the IPSSP method is based on the estimation of frequencies, we used a reduced version of the dependency model originally employed by the IPSSP method (see Aydin *et al.* [24] for details). We call this simplified version as IPSSP-simp. We applied the composition based reduction and used a threshold of 0.35 in the dataset reduction step (see Section 2.3.1.1). We also used the Laplace's rule as the pseudo-count method to initialize the frequency tables, in which each entry is originally set to one. With these parameters, the sensitivity measures of the IPSSP-simp method are obtained as follows: $Q_3(\%) = 67.95\%$, $Q_\alpha^{obs}(\%) = 68.42\%$, $Q_\beta^{obs}(\%) = 48.72\%$, $Q_L^{obs}(\%) = 76.57\%$. From these results, we can conclude that IPSSP is a powerful single-sequence method and is capable of producing state-of-the-art results.

2.5.3 Contribution of the Model Components

2.5.3.1 Contribution of Dependency Models

In this section, we assess the individual performances of the dependency models used by the IPSSP algorithm. We compare the performances of $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, and $\mathcal{M}_C$, where $\mathcal{M}_C$ is the combined model (see Section 2.2.4). We also include the performance of the IPSSP algorithm when no dataset reduction scheme is applied (denoted by PSSP-$\mathcal{M}_C$). The results in Table 24 show that the combined model improve the overall accuracy of the IPSSP method by more than 1%. In this experiment, the second conversion rule (H, G to H, E, B to E and all other states to L) is used without length adjustments and all three models use a five letter alphabet for positions with significantly high correlation measures. The performance obtained when all the hydrophobicity groupings are defined using the three letter alphabet is 0.4% lower (data not shown). Therefore, as compared to the BSPSS method, the dependency models improve the overall sensitivity measure by 1.6%. The inclusion of the training set reduction further improves the accuracy by 0.5%.

Table 24: Performances of the BSPSS, IPSSP with dependency models, $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, and IPSSP with the combined model, $\mathcal{M}_C$ (obtained using an averaging filter), evaluated on the EVA set under the single-sequence condition.

Sensitivity	$Q_3(\%)$	$Q_\alpha(\%)$	$Q_\beta(\%)$	$Q_L(\%)$
BSPSS	65.175	65.640	38.814	76.658
IPSSP-$\mathcal{M}_1$	65.968	66.199	45.387	75.043
IPSSP-$\mathcal{M}_2$	66.003	66.606	**46.952**	74.108
IPSSP-$\mathcal{M}_3$	66.315	67.012	45.005	75.364
IPSSP-$\mathcal{M}_C$	**67.421**	**68.089**	46.395	76.363
PSSP-$\mathcal{M}_C$	66.840	66.945	44.566	**76.761**

2.5.3.2 Contribution of the Dataset Reduction

In this section, we compare the training set reduction techniques described in Section 2.3.1. In our simulations, we used the EVA set of "sequence-unique" proteins. To reduce eight secondary structure states to three, we used the following conversion

rule: H, G to H; E, B to E; I, S, T, ' ' to L. We used the PDB_SELECT dataset to compute the Chou-Fasman coefficients (*i.e.*, the function $f(.)$ in Eq. (25) and Eq. (26)) described in [42] (see Section 2.5.1 for details of the datasets). Here, the coefficients reflect the propensity of an amino acid to be either in H, E, or L state, which are defined in Section 2.3.1.3.

We evaluated the performances of the methods by a leave-one-out cross validation experiment (jacknife procedure). To expedite the evaluations, we restricted only our test data to the first 600 proteins in the dataset, which gave a good approximation to the true result. We chose the three-state-per-residue accuracy, Q_3, as the overall sensitivity measure (see Section 2.5.1.2).

Tables 25 and 26 show the performance of the IPSSP method with respect to various training set reduction schemes. Table 25 summarizes the results when the first 80% of the most similar proteins is selected and Table 26 provides the accuracy measures for the threshold-based reduction. Among the reduction schemes being compared in Table 25, the composition based reduction is the most accurate. This is mainly because of the fact that composition based reduction does not impose strong constraints, which compensates for the errors made in the initial secondary structure prediction. In addition, threshold based reduction is slightly better than the reduction that selects the first 80% of the most similar proteins. Hence, the composition based reduction method with thresholding gave the best performance, where the secondary structure prediction accuracy is improved by 0.6% compared to the condition with no re-training. Another advantage of the composition based method is its low computational complexity.

Comparing the alignment based reduction methods, the best result is obtained by the method that aligns secondary structures. Joint alignments of amino acid sequences and secondary structures did not perform better than secondary structure alignments. This is not surprising because in single-sequence condition the input

protein is not statistically similar to dataset proteins at the amino acid level. Therefore, the discriminative power of the amino acid similarity matrix is weaker than the secondary structure similarity matrix.

Table 25: Sensitivity measures of the training set reduction methods employed in the IPSSP algorithm. The top 80% of the proteins are classified as similar to the input protein.

Method	$Q_3(\%)$
Composition Based	67.01
Alignment Based (SS)	67.00
Alignment Based (AA+SS)	66.92
Alignment Based (AA)	66.69
Chou-Fasman Based (D_{cf})	66.65
No Re-training	66.59
Chou-Fasman Based ($D_{cf,2}$)	66.50

Table 26: Sensitivity measures of the training set reduction methods employed in the IPSSP method. The dataset proteins are classified as similar to the input protein by applying a threshold.

Method	$Q_3(\%)$
Composition Based	67.17
Alignment Based (SS)	67.12
Alignment Based (AA+SS)	67.06
No Re-training	66.59

2.5.4 Conversion Rules, Length Adjustments and Prediction Confidence

2.5.4.1 Conversion Rules and Length Adjustments

To investigate the effect of length adjustments, we considered converting short α-helices and β-strands to loops so that the α-helix and β-strand segments had at least five and three residues, respectively (see Section 2.5.1.4). Then, we compared IPSSP and BSPSS using different conversion rules and length adjustments. As seen in Table 27, IPSSP performs better than BSPSS for each set of rules.

Table 27: Prediction sensitivity measures, $Q_{.}(\%)$, analyzed with respect to three conversion rules and length adjustments, evaluated on the EVA set under the single-sequence condition.

Sensitivity	$Q_3(\%)$	$Q_\alpha(\%)$	$Q_\beta(\%)$	$Q_L(\%)$
BSPSS Rule 1	65.177	65.655	38.844	76.644
BSPSS Rule 2	65.175	65.640	38.814	76.658
BSPSS Rule 3	67.218	64.048	38.071	80.491
BSPSS Rule 1 + Length adj	68.060	63.775	37.022	81.378
BSPSS Rule 2 + Length adj	68.078	63.793	37.017	81.399
BSPSS Rule 3 + Length adj	68.400	63.203	36.737	**82.167**
IPSSP Rule 1	67.415	**68.115**	46.386	76.340
IPSSP Rule 2	67.421	68.089	**46.395**	76.363
IPSSP Rule 3	69.096	66.559	45.319	79.893
IPSSP Rule 1 + Length adj	70.027	66.557	45.588	80.577
IPSSP Rule 2 + Length adj	70.036	66.554	45.559	80.602
IPSSP Rule 3 + Length adj	**70.300**	65.934	45.445	81.280

2.5.4.2 Prediction Confidence

To estimate the confidence in predictions we computed the overall sensitivity, Q_3, as a function of the probability assigned to the predicted state at each position (see Figure 7). For instance, at a threshold prediction probability of 0.6, with 54% of sequence positions in this category, we achieved a Q_3 of 78.5%. On the other hand, at a threshold prediction probability of 0.8, with 16% of positions in this category, we obtained a Q_3 of 89.5% as shown in Table 28. In terms of the prediction confidence and the total number of the positions covered, IPSSP is comparable to PSIPRED and better than BSPSS at all prediction thresholds.

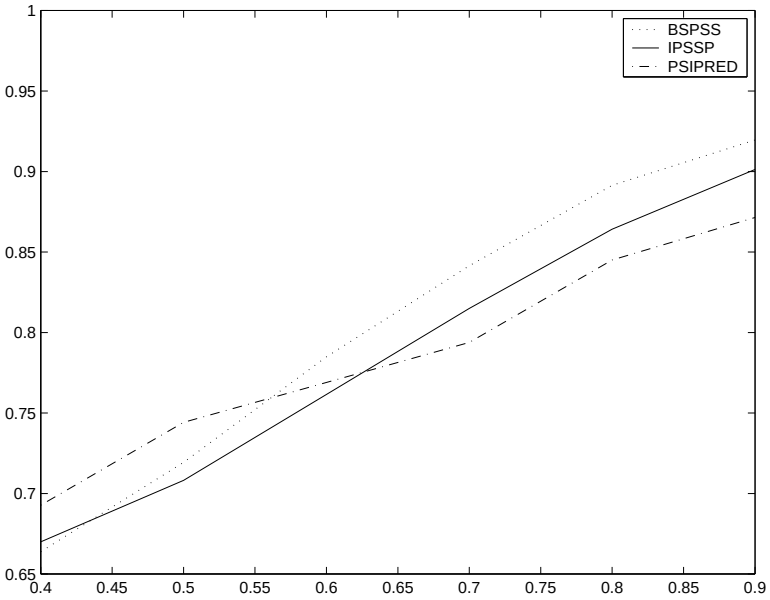


Figure 7: Prediction confidence values vs prediction threshold.

Table 28: Percentage of true positives for predictions made in a set of positions having the *a posteriori* probability of the predicted state above the threshold. To reduce eight states to three, the second conversion rule (H, G to H, E, B to E and all other states to L) is used.

Prediction Confidence (% Positions)			
Prediction Threshold	0.4	0.6	0.8
BSPSS	0.661 (95.45)	0.781 (48.14)	0.889 (12.52)
IPSSP	**0.676 (96.71)**	**0.785 (54.38)**	**0.895 (16.82)**

2.6 Summary

In this chapter, we showed that new dependency models and training methods bring further improvements to protein secondary structure prediction in single-sequence setting. The results are obtained under cross-validation conditions using a dataset with no pair of sequences having significant sequence similarity.

The improvements over the BSPSS method [134], which also employs hidden semi-Markov models can be summarized as follows. We introduced three residue dependency models (both probabilistic and heuristic) incorporating the statistically

significant amino acid correlation patterns at structural segment borders. In those models, we allowed dependencies to positions outside the segments to relax the condition of segment independence. Another novelty of the models is the dependency to downstream positions, which we believe is necessary due to asymmetric correlation patterns observed uniformly in structural segments. Apart from the more elaborate dependency structure, we introduced a training set reduction strategy to refine estimates of the model parameters. Among the dataset reduction methods, the composition based reduction technique with thresholding generated the most accurate results. This is mainly because of the fact that composition based reduction does not impose strong constraints, which serves to compensate for the errors made in the initial secondary structure prediction.

Typically protein secondary structure prediction methods suffer from low accuracy in predicting β-strands, in which non-local correlations have a significant role. In this chapter, we did not specifically address this problem, but showed that improvements are possible when higher order dependency models are used and significant correlations outside the segments are considered.

CHAPTER III

BAYESIAN PROTEIN SECONDARY STRUCTURE PREDICTION WITH NEAR-OPTIMAL SEGMENTATIONS

3.1 Introduction

Secondary structure prediction is an invaluable tool in determining the three-dimensional structure and the function of proteins. Typically, protein secondary structure prediction methods suffer from low accuracy in β-strand predictions, where non-local interactions play a significant role [134, 45, 44, 58, 23]. The β-strand sensitivity of a single-sequence prediction method is approximately 25-50% and that of a method using evolutionary information is between 50-65%. The low accuracy of β-strand predictions is mainly because of the difficulty in modeling non-local interactions that are characteristic of β-strands. For instance, the Bayesian inference approach and the hidden semi-Markov model introduced in Chapter 2 has some limitations due to the assumptions made in the model derivation. We assumed that the segment likelihood terms are independent from each other as formulated in Eq. 5. This assumption enabled us to implement efficient hidden Markov models. However, with this assumption and others inherent in the theory of hidden Markov models, it is not possible to model the long-range interactions that have a significant role in the stabilization of the 3-D structure.

In single-sequence predictions, Frishman and Argos [58] proposed a method that incorporates a non-local interaction model into a nearest-neighbor algorithm. Their method achieved an overall accuracy of 68%, which is not significantly higher than the accuracy of the current state-of-the-art methods utilizing local correlations only [23].

Besides, for longer protein sequences with many potential stretches of β-strand residues, the mutual signal from complementary β-strands fades and even the distinction between anti-parallel and parallel sheets becomes weak. Chu *et al.* [45, 44] and Cheng and Baldi [39] combined multiple alignment profiles with non-local interaction models. Chu *et al.* [45, 44], extended the work by Schmidler *et al.* [134, 135] and incorporated the multiple alignment sequence profiles into the semi-Markov model. They achieved an overall sensitivity of 72-74% and a β-strand sensitivity of 56-59% from a local dependency model with multiple alignment profiles. However, they did not report any improvement in secondary structure prediction accuracy through the incorporation of non-local interactions. Moreover, their model is based on β-strand segment pair propensities and does not impose global constraints for β-sheet formation. Cheng and Baldi [39] proposed a three-stage modular approach to predict and assemble the β-sheets of a native protein. Their method exploits global covariation and constraints characteristic of β-sheet architectures and achieves significant improvements over the existing methods in predicting the β-strand pairs, the interaction types (parallel, anti-parallel), and interactions at the amino acid level (*i.e.*, contact maps). However, they assume that the true secondary structure segmentation is available (either as an experimental sequence or as a prediction) and then find the optimum β-sheet conformation for that segmentation. They did not analyze how the derived energy functions can discriminate false secondary structure segmentations from the correct one, and did not apply their method to the problem of secondary structure prediction. Therefore, there is still a considerable need to model long-range interactions that contribute to the stabilization of a protein molecule in an attempt to improve the accuracy of the secondary structure prediction.

In this chapter, we introduce an alternative decoding technique for the hidden semi-Markov model originally introduced in Chapter 2 (see also Aydin *et al.* [23], Schmidler *et al.* [134], and Chu *et al.* [44]). The proposed method is based on the

N-best paradigm where a set of suboptimal segmentations (*N-best list*) is computed as an alternative to the most likely segmentation. N-best methods have found diverse applications in speech recognition [136, 137, 110, 139], sequence-sequence alignments [147, 133, 72], sequence-structure alignments [99, 31], gene prediction [87, 37], and topology prediction for outer-membrane proteins [55, 26]. To compute suboptimal segmentations, we developed two N-best algorithms: modified stack decoder and N-best Viterbi. The first one is an A^* stack decoder algorithm that extends paths (or hypotheses) by one symbol at each iteration. The second algorithm locally keeps the end positions of the highest scoring K previous segments and performs backtracking. Both algorithms employ the hidden semi-Markov model described in Chapter 2 and use Viterbi scoring to compute the *N-best list*. The availability of near-optimal segmentations and the utilization of the Viterbi scoring enable the sequences to be re-scored by more complex dependency models that characterize non-local interactions in β-sheets. After the score update, one can either keep the segmentations to be employed in 3-D structure prediction or compute a secondary structure prediction by applying a weighted voting procedure to a set of top scoring $M \geq 1$ segmentations.

3.2 Generating an N-best List

There are a few methods in the literature that compute an N-best list. These algorithms can be based on N-best search (*e.g.*, time-synchronous Viterbi style beam search) [136, 137], an A^* search [110], tree-trellis approach [139], or on divide and conquer methods [106]. Different from the Viterbi algorithm, which finds the most likely state sequence (or path), an N-best method finds the most likely labeling of a given sequence as well as suboptimal labelings (or segmentations). Note that in many applications [87, 26, 137], there can be more than one state sequence that contribute to the same labeling of a given sequence. Therefore in general, an N-best algorithm always produces a labeling with a probability at least as high as the result of the

Viterbi algorithm. In secondary structure prediction, however, there is a one-to-one correspondence between a state sequence and a labeling. In other words, there can be only one state sequence per labeling. Hence, an exact N-best algorithm will produce the Viterbi segmentation as the most likely secondary structure labeling, and the 1-Best procedure described in [87] reduces to the Viterbi algorithm.

In this chapter, we develop two approximate N-best algorithms for protein secondary structure prediction that employ hidden semi-Markov models. The first algorithm is a modified stack decoder and the second one is an extension of the Viterbi search. In the next section, we will describe the modified stack decoder algorithm.

3.2.1 Modified Stack Decoder

Stack decoder, a search methodology well-known in the speech recognition literature, was introduced by researchers at IBM [76], and is a variant of the A^* search [110, 107]. One can think of a stack decoder as a sub-optimal tree search with many appealing properties. The basic stack decoder algorithm can be found in [110]. The ideas underpinning stack decoding are those of sequential decoding in communications theory [76] and of heuristic search in artificial intelligence [107]. These search algorithms are time asynchronous, in which the best scoring path or hypothesis, irrespective of time, is chosen for extension and this process is continued until a complete hypothesis is determined. In the classical implementation of the stack decoder, the stack consists of an ordered heap, which holds a number of partial hypotheses, (*e.g.* partial secondary structure labelings). At each iteration, hypotheses of different lengths are extended by one segment and are compared to each other, where only the high scoring ones are kept in the stack as surviving paths.

The crucial function for a stack decoder algorithm is the estimated score (log likelihood) of the hypothesis h at time t, and is given by:

$$f_h(t) = a_h(t) + b_h^*(t), \tag{34}$$

where $a_h(t)$ is the score of the partial hypothesis using information up to time t and $b_h^*(t)$ is the estimate of the best possible score (maximum log likelihood) in extending the partial hypothesis to a valid complete hypothesis. It has been shown that as long as $b_h^*(t)$ is an upper bound on the actual log likelihood, then the search algorithm is admissible [107] (*i.e.*, no errors will be introduced that would not occur if an exhaustive search was performed). This approach allows the hypotheses of different lengths to be compared. However, the disadvantage of approximating $b_h^*(t)$ is the requirement to look ahead at the data. An alternative approach [27, 63, 110], does not rely on looking ahead. Instead, $b_h^*(t)$ is constructed such that hypotheses with earlier reference times always have higher scores than those with later reference times.

In this section, we propose a modified stack decoder algorithm to generate suboptimal segmentations of secondary structure for a given amino acid sequence. Our approach is similar to the Tailbiting decoder introduced in [18]. In the proposed method, each hypothesis of the stack consists of a secondary structure sequence extended up to position j, where $1 \leq j \leq n$, and n is the total length of the amino acid sequence. The score of the i^{th} hypothesis with length j is defined as $P(\mathbf{R}_{[1:j]}, \mathbf{S}_j^{(i)}, \mathbf{T}_j^{(i)})$, which is the joint probability of observing the amino acid sequence up to position j ($\mathbf{R}_{[1:j]}$), and the secondary structure labeling of the hypothesis ($\mathbf{S}_j^{(i)}, \mathbf{T}_j^{(i)}$). Here, $1 \leq i \leq N$, where N is the stack size.

The steps of the algorithm is as follows. We first initialize the stack by including all possible segmentations up to a certain position (j^*) so that the stack contains exactly N segmentations. Then, for each hypothesis, we consider possible candidate extensions and keep the ones with the highest scores. Here, an extension is obtained by concatenating a single secondary structure *symbol* (either H, E, or L) instead of a secondary structure segment. At each iteration, we extend the hypotheses by one symbol until the n^{th} position is reached, so that each hypothesis consists of a secondary structure sequence of length n. Finally, we sort the hypotheses in decreasing order

of scores. Stack initialization and hypothesis extension steps of the algorithm are illustrated in Figure 8. Since an extension is performed by concatenating a single

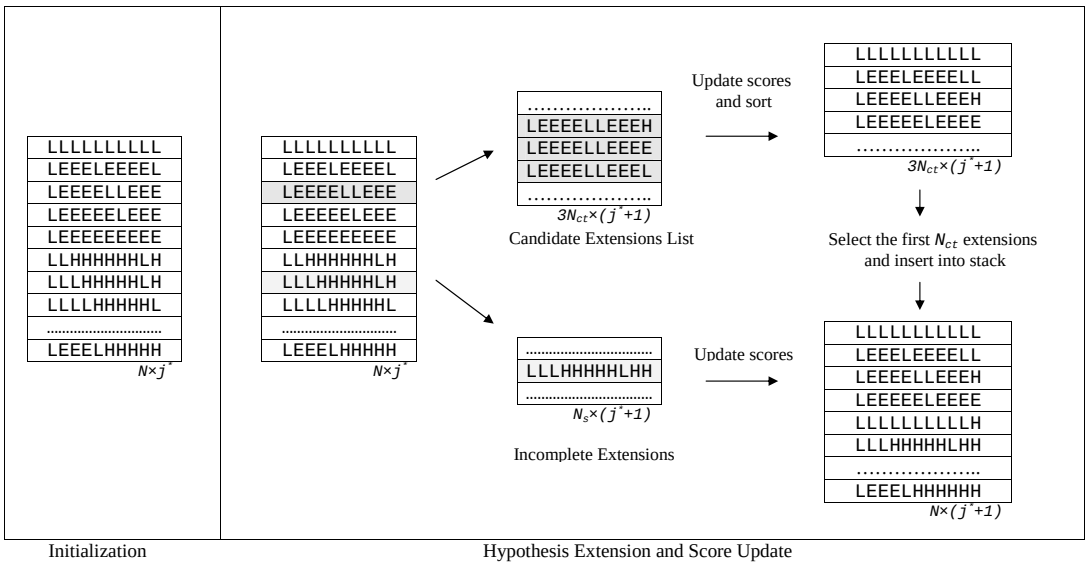


Figure 8: The modified stack decoder algorithm.

secondary structure symbol instead of a segment, at a given iteration, each hypothesis has the same length. This approach ensures fair comparisons between the scores of the individual hypotheses, and eliminates the need to approximate or construct $b_h^*(t)$ in Eq. (34). Another advantage of this method is related to the selection of the best extension for a given hypothesis. In the case of segment extensions, we are most likely to choose the segments with minimum lengths because for local extensions, shorter segments have higher probability scores. One way to solve this problem would be to design a score normalization method to compensate for the decrease in the score of a hypothesis due to its length. Unfortunately, such methods usually hinge on some kind of a heuristic, which may not perform well for different protein families. Therefore, we are proposing a method that extends the hypotheses by one symbol at each iteration.

The selection of the best scoring extensions from position j to $j+1$ is as follows. We first obtain the list of all possible candidate extensions derived from the entire

set of hypotheses[1]. In computing the extensions, we satisfy the minimum length requirements for the three types of secondary structure. In the current implementation, we restricted the lengths of the α-helices, β-strands, and loops to be greater than or equal to 5, 3, and 1 respectively. Before extending a hypothesis, we first check if the last secondary structure segment in the hypothesis satisfies the minimum length requirement. If the length of the last segment is already greater than or equal to the corresponding lower bound, then all three extensions (H, E and L) are performed and the extended hypothesis is stored in the candidate extension list. If the last segment is shorter than the lower bound, then that segment is extended only by its existing secondary structure type and that hypothesis is kept in the stack without being included in the candidate extensions list. If the number of such hypotheses with incomplete extensions is N_s, then the number of hypotheses that are extended and included into the candidate extensions list becomes $N_{ct} = N - N_s$, and the total number of hypotheses in the candidate extension list becomes $N_{ce} = 3N_{ct}$. Hence, the set of candidate extensions is derived from those hypotheses, in which all secondary structure segments satisfy the minimum length requirements. Having compiled the list of candidate extensions, we compute the score of each hypothesis using the parameters of the hidden semi-Markov model. Finally, we sort the hypotheses in the candidate extension list in decreasing order of scores and insert the first N_{ct} hypotheses back into the stack. Note that for a hypothesis in the candidate extension list, if the extension initiates a new α-helix or β-strand segment, then this extended hypothesis will not satisfy the minimum length requirement. To prevent the score of the new hypothesis to be computed as zero[2], we modified the length distribution of the α-helices and β-strands for small segments to take non-zero values. We chose a

[1]Maximum length of the list is $3 \times N$, where N is the total number of hypotheses or the stack size.

[2]Parameter estimation for hidden semi-Markov model was initially performed using maximum-likelihood estimation procedure on a training set, in which each protein satisfies the minimum length requirements.

value that is large enough to initiate α-helix and β-strand segments and small enough to avoid paths with dominantly short segments. In the current implementation, the probability of short α-helices ($l_H < 5$) and short β-strands ($l_E < 3$) is set to 10^{-5}. The steps of the algorithm is summarized in Algorithm 2.

Algorithm 2: Modified Stack Decoder Algorithm

Input: Amino acid sequence **R**, Stack of size N, Candidate Extension List of size $3N$

Output: Secondary Structure Prediction **SS**

1 Initialize the stack of size N with all possible extensions up to position $j = j^*$;
2 $j \leftarrow j + 1$;
3 **repeat**
4 Select a hypothesis from the list;
5 **if** *last segment is shorter than the length threshold* **then**
6 Extend the hypothesis only with the type of the last segment;
7 Keep the extended hypothesis in the stack;
8 **else**
9 Perform all possible extensions (H, E, L);
10 Put the extended hypotheses into the candidate extensions list;
11 Delete the original hypothesis from the stack;
12 **until** *all N hypotheses are extended*;
13 Number of hypotheses in stack = N_s;
14 Number of hypotheses in the candidate extensions list = $3(N - N_s)$;
15 Select the top $N - N_s$ hypotheses from the candidate extension list;
16 Insert into stack;
17 **if** $j = n$ **then**
18 End of sequence is reached. Terminate;
19 **else**
20 Go to step 2;

To evaluate the computational complexity of the algorithm, it is useful to divide the operations into two parts: (i) sorting, (ii) score computation. To obtain the top scoring $N - N_s$ hypotheses in the candidate extensions list, we use the heap sort algorithm, which has $O(K \log K)$ complexity, where K is the size of the list that is going to be sorted. In C implementation, it takes approximately 30 seconds to sort a list of 10^6 hypotheses using the heap sort algorithm. Since sorting operations are performed for each position $j = j^* + 1, ..., n$, the total number of such operations is

$n\overline{N_{ce}} \log \overline{N_{ce}}$, where $\overline{N_{ce}}$ is the average size of the candidate extensions list. In the worst case scenario, $\overline{N_{ce}}$ takes the value $3N$. Therefore, the computational requirements of the sorting operations is on the order of $O(nN \log N)$. The computational complexity arising from the score computation is on the order of $O(nN)$. For a protein of length 200 amino acids, and a stack of size $N = 30,000$, it takes approximately five minutes to perform all the extensions up to the last position and obtain a sorted list. Here, the algorithm is tested on an Intel Pentium III Processor with a 1.2GHz CPU and a 512MB RAM.

3.2.2 N-best Viterbi Algorithm

As an alternative approach, we developed the N-best Viterbi algorithm, which is a generalization of the classical Viterbi algorithm. The idea is analogous to the Word-Dependent N-best algorithm introduced by Schwartz and Austin [136]. In the classical Viterbi algorithm, for each secondary structure segment that is of type $t \in \{H, E, L\}$ and ends at position j, we consider possible previous segments that are of type $l \neq t$ and end at position v. We then store the maximum value of the score function $f(.)$, and the arguments (v, l) where that maximum is achieved. The definition of the score function $f(.)$ is as follows:

$$\begin{aligned} f(v,l,j,t) &= \delta(v,l) P(T = t \mid T_{prev} = l) \\ &\times P(S = j \mid T = t, S_{prev} = v) \\ &\times P(R_{[v+1:j]} \mid S_{prev} = v, S = j, T = t) \\ \delta(j,t) &= \max_{v,l} f(v,l,j,t) \end{aligned} \tag{35}$$

In the above formulation, $\delta(v, l)$ is the joint probability of observing the amino acid sequence and the secondary structure labeling from position 1 to v. Here, the secondary structure sequence labeling is the maximum scoring path from position 1 to v, in which the last segment is of type l. The algorithm iterates for positions $j = 1, ..., n$,

where n is the total number of amino acids in the protein and v can take the values from 1 to $j-1$.

In the N-best Viterbi algorithm, for each (j,t), instead of storing the maximum value and the arguments of $f(.)$, we rank the possible values of this function with respect to (v,l) and store the highest scoring K local values as well as the arguments where these values are achieved. Here, K typically takes values from 3 to 6. The difference of this approach from the well known N-best algorithm [137, 26, 87] is that at each state t ending at position j, position indices and types of the K local previous segments are stored instead of the all segment histories (or paths) ending at that position. The recursion for the forward pass can be formulated as follows:

$$\begin{aligned}
\delta_k(j,t) &= rank_k((v,l), f(v,l,j,t)) \qquad (36)\\
&= rank_k((v,l), \delta_1(v,l)P(T=t \mid T_{prev}=l))\\
&\times \; P(S=j \mid T=t, S_{prev}=v)P(R_{[v+1:j]} \mid S_{prev}=v, S=j, T=t)\\
\psi_k^P(j,t) &= \arg rank_k(v, f(v,l,j,t))\\
\psi_k^T(j,t) &= \arg rank_k(l, f(v,l,j,t))\\
k &= 1,...,K.
\end{aligned}$$

Here, $rank_k(x,g)$ outputs the k^{th} value of the function $g(.)$ with respect to the argument set x, where $k=1,..,K$. Similarly, $\arg rank_k(x,g)$ returns the the argument set x, where the k^{th} value of $g(.)$ is achieved. $\delta_k(j,t)$ is the joint probability of observing the amino acid sequence and the secondary structure labeling $(\mathbf{S}_j, \mathbf{T}_j)_k$ from position 1 to j. Note that $(\mathbf{S}_j, \mathbf{T}_j)_k$ does not necessarily correspond to the k^{th} best path[3] from position 1 to j. Instead, it defines a path that satisfies the following constraints: (1) The last secondary structure segment is of type t and ends at position j; (2) The segment before the last segment is of type l_k and ends at position v_k; (3) The segment before the last segment is on the maximum scoring path that ends at v_k

[3]The k^{th} path is guaranteed for $k=1$.

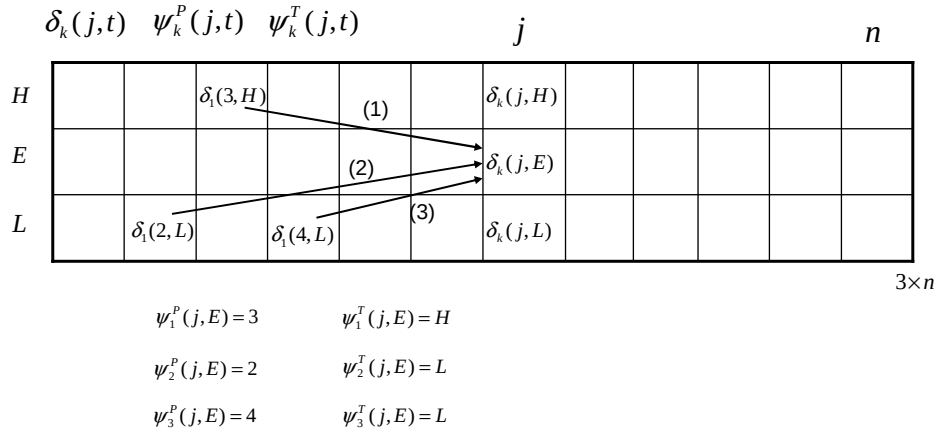


Figure 9: The forward pass of the N-best Viterbi algorithm.

with a secondary structure type l_k different from t. The arguments v_k and l_k, where the $f(v,l,j,t)$ takes its k^{th} value are stored into $\psi_k^P(j,t)$ and $\psi_k^T(j,t)$, respectively. An iteration of the forward pass is described in Figure 4.

Once the forward pass is completed, we perform backtracking and generate alternative prediction sequences. We start with the n^{th} position and consider all $3K$ segments (K segments for each secondary structure type) that end at this position and are of length $n - v_k$, where v_k is the end position of the previous segment that was stored in the forward pass. We insert these hypotheses into an array of size N and represent them by character strings, in which the first v_k values are set to 'X' and the last $n - v_k$ values are assigned to the secondary structure type of the last segment. Then, for each hypothesis in the array, we perform all possible extensions by one segment in the right-to-left direction, and replace the existing hypotheses with the extended versions. Note that since two adjacent segments cannot be the same in a hidden semi-Markov model, the total number of extensions for each hypothesis is $2K$. If the array becomes full before all the hypotheses are extended up to the first position, then we keep those sequences that are already extended completely and extend only the non-complete sequences. This time, the extensions are performed according to the maximum scoring paths. We terminate when all N sequences are

extended up to the first position. The algorithm is summarized in Algorithm 3.

Algorithm 3: N-best Viterbi Algorithm

Input: Amino acid sequence $\mathbf{R}$, Hidden semi-Markov Model, Array of size $N \times n$

Output: N-best list of secondary structure segmentations

```
1  for each position and secondary structure type do
2      Locally keep the end positions of the highest scoring K previous segments;
3      Insert the 3K segments that end at position n into the array of size N;
4      array-full flag = FALSE;
5      repeat
6          for i^th hypothesis in the array do
7              if array-full flag = FALSE AND extension-finished flag_i = FALSE then
8                  Perform 2K back-extensions (add segments in the right-to-left direction), in
                   which the previous segment types are different from the type of the current
                   segment;
9                  for each back-extension do
10                     if # hypotheses = N then
11                         array-full flag = TRUE;
12                     else
13                         Insert the extended hypothesis into the array;
14                     if back-extended hypothesis reaches the N-Terminal of the protein then
15                         extension-finished flag_i = TRUE;
16             else
17                 if array-full flag = TRUE AND extension-finished flag_i = FALSE then
18                     Perform the maximum scoring extensions only;
19                     if back-extended hypothesis reaches the N-Terminal of the protein then
20                         extension-finished flag_i = TRUE;
21                 else
22                     continue with the next hypothesis;
23     until array-full flag = TRUE AND extension-finished flag_i = TRUE, 1 ≤ i ≤ N;
24     Sort the hypotheses;
25     Terminate;
```

The computational complexity of the algorithm can be evaluated as follows. In

the forward pass, for each position j and secondary structure type t that represents a secondary structure segment ending at position j, the highest scoring K local paths are computed. To do this, we need to consider the segmentations such that the end position of the previous segment, v, takes values from 1 to $j-1$. There are a total of $(n-1)n/2$ such segmentations for $j = 1, .., n$. This requires $3K(n-1)n/2$ operations. Hence, the computational complexity of the forward pass from score computations is $O(Kn^2)$. At each position j and secondary structure type t, we keep two arrays of size K to store the segment end position of the previous segments and the corresponding path scores. To keep K previous segment end positions, a total of $2 \times 3K \times n$ comparisons are required. Hence, the computational requirement to keep K local paths is $O(Kn)$. Backtracking can be performed by a fast recursive procedure or an iterative approach, which takes $O(Kn)$ back extensions. Finally, the sequences are sorted by a heap sort algorithm, with $O(N \log N)$ complexity. For a protein of length 200 amino acids, and a stack of size $N = 30,000$, it takes approximately one minute to obtain the sorted list of N suboptimal sequences. Though the N-best Viterbi algorithm is faster than the modified stack decoder algorithm for a given N value, the list obtained by the N-best Viterbi algorithm contains more segmentations that are similar to each other than the list delivered by the stack decoder. In other words, the stack decoder algorithm generates a deeper list for a given list size and one needs to increase N in the N-best Viterbi algorithm to be able to reach the same depth level.

3.2.3 HSMM Implementation Details

We developed two N-best algorithms that generate suboptimal segmentations from hidden semi-Markov models. In HSMM implementation, we used alternative versions of the dependency patterns introduced in Section 2.2.3. Since the CB513 set contains less number of proteins than the EVA set, we further reduced the dependencies in

Table 10 to reliably estimate model parameters in a cross validation experiment. Tables 29 and 30 show the dependency patterns (feature sets) derived for the EVA and CB513 sets, respectively. Those tables are employed by the Viterbi and N-best algorithms. Similarly, Tables 31 and 32 show the dependency models used for the IPSSP algorithm. Those tables are employed by the IPSSP method, which is used to derive the marginal *a posteriori* distribution for the score update (see Section 3.4.1).

The tables showing the dependency patterns are divided into panels for α-helix, β-strand, and loop segments. We identify as terminal positions (N1-N4, C1-C4) those in which the amino acid frequency distributions significantly deviate from ones in internal positions (Int) in terms of the Kullback-Liebler (KL) distance [134]. Here, N1 represents the first and C1 represents the last position of a secondary structure segment. Based on the available training data, we chose six proximal positions (N1-N4, C1-C2) for α-helices, four proximal positions (N1-N2, C1-C2) for β-strands, and eight proximal positions (N1-N4, C1-C4) for loops. The remaining positions are defined as internal positions (Int).

To reduce the dimension of the parameter space, we grouped the amino acids into three hydrophobicity classes. For instance, h^3_{i-2} stands for the dependency of the amino acid at position i to the hydrophobicity class of the amino acid at position $i-2$. The superscript 3 represents the total number of hydrophobicity classes. The probability of observing an amino acid at a given position is then defined using the dependence patterns for that position only. From Table 29, the conditional probability of observing an amino acid at position $i = N3$ of an α-helix segment is $P_{N_3}(R_i \mid h^3_{i-2}, h^3_{i-3}, h^3_{i+2})$. Finally, for each dependency model, the segment likelihood term $P(\mathbf{R}_{[S_{j-1}+1:S_j]} \mid S_{j-1}, S_j, T_j)$ is computed by multiplying the conditional probabilities selected from the corresponding table. In addition to position specific dependencies, we derived separate patterns for segments with different lengths. All tables show the dependence patterns for segments longer than L residues, where L is

five for α-helices, and three for β-strands and loops. For shorter segments, we selected a representative set of patterns based on the available training data (not shown). A more detailed description of the dependency models can be found in Chapter 2.

Table 29: Positional dependencies within structural segments for the Viterbi and N-best list algorithms evaluated on the EVA set. $h_j^3 \in \{hydrophobic, neutral, hydrohilic\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$.

α-helix							β-strand				
Int	N1	N2	N3	N4	C1	C2	Int	N1	N2	C1	C2
h_{i-2}^3	h_{i-1}^3	h_{i-2}^3	h_{i-2}^3	h_{i-2}^3	h_{i-2}^3	h_{i-2}^3	h_{i-1}^3	h_{i-1}^3	h_{i-1}^3	h_{i-1}^3	h_{i-1}^3
h_{i-3}^3	h_{i+2}^3	h_{i+2}^3	h_{i-3}^3	h_{i+2}^3	h_{i-4}^3	h_{i-4}^3	h_{i-2}^3	h_{i-2}^3	h_{i-2}^3	h_{i-2}^3	h_{i-2}^3
h_{i-4}^3	h_{i+4}^3	h_{i+4}^3	h_{i+2}^3	h_{i+4}^3	h_{i+1}^3	h_{i+1}^3	h_{i+1}^3	h_{i+1}^3	h_{i+2}^3	h_{i+2}^3	h_{i+1}^3
h_{i+2}^3							h_{i+2}^3				
h_{i+4}^3											

Loop					
Int	N1	N2,N3,N4	C1	C2,C3	C4
h_{i-1}^3	h_{i-1}^3	h_{i-1}^3	h_{i-1}^3	h_{i-1}^3	h_{i-2}^3
h_{i-2}^3	h_{i-2}^3	h_{i-2}^3	h_{i-2}^3	h_{i+1}^3	h_{i-3}^3
h_{i-3}^3	h_{i+1}^3	h_{i-3}^3	h_{i+1}^3	h_{i+2}^3	h_{i+1}^3
h_{i+1}^3	h_{i+3}^3	h_{i+1}^3	h_{i+2}^3	h_{i+3}^3	h_{i+2}^3
h_{i+2}^3					

Table 30: Positional dependencies within structural segments for the Viterbi and N-best list algorithms evaluated on the CB513 set. $h_j^3 \in \{hydrophobic, neutral, hydrohilic\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$.

α-helix				β-strand		Loop	
Int	N1	N2,N4,C1-2	N3	Int	N1-2, C1-2	Int	N1-4, C1-4
h_{i-2}^3	h_{i-1}^3	h_{i-2}^3	h_{i-3}^3	h_{i-1}^3	h_{i-1}^3	h_{i-1}^3	h_{i-1}^3
h_{i-3}^3				h_{i-2}^3		h_{i-2}^3	h_{i+1}^3
h_{i-4}^3						h_{i-3}^3	

Table 31: Positional dependencies within structural segments for the models $\mathcal{M}1$, $\mathcal{M}2$, and $\mathcal{M}3$ of the IPSSP method evaluated on the EVA set. $h_j^3 \in \{$*hydrophobic*, *neutral*, *hydrohilic*$\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$. h_j^5 is a 5 letter alphabet with groups defined as $\{P, G\}$, $\{E, K, R, Q\}$, $\{D, S, N, T, H, C\}$, $\{I, V, W, Y, F\}$, $\{A, L, M\}$.

		$\mathcal{M}1$	$\mathcal{M}2$	$\mathcal{M}3$
H	Int	$h_{i-2}^5, h_{i-3}^3, h_{i-4}^5, h_{i-7}^3$	$h_{i-2}^3, h_{i-3}^3, h_{i-4}^3, h_{i+2}^3, h_{i+4}^3$	$h_{i+2}^5, h_{i+3}^5, h_{i+4}^5$
	N1	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+2}^5	h_{i+2}^5, h_{i+4}^5
	N2	$h_{i-1}^3, h_{i-2}^3, h_{i-3}^3$	$h_{i-2}^3, h_{i+2}^3, h_{i+4}^3$	$h_{i+1}^3, h_{i+2}^3, h_{i+4}^3$
	N3	$h_{i-1}^3, h_{i-2}^3, h_{i-3}^3$	$h_{i-2}^3, h_{i-3}^3, h_{i+2}^3$	$h_{i+1}^3, h_{i+2}^3, h_{i+4}^3$
	N4	$h_{i-1}^3, h_{i-2}^3, h_{i-3}^3$	$h_{i-1}^3, h_{i+2}^3, h_{i+4}^3$	$h_{i+1}^3, h_{i+2}^3, h_{i+4}^3$
	C1	$h_{i-1}^3, h_{i-2}^3, h_{i-4}^3$	$h_{i-2}^3, h_{i-4}^3, h_{i+1}^3$	$h_{i+1}^3, h_{i+2}^3, h_{i+3}^3$
	C2	$h_{i-2}^3, h_{i-3}^3, h_{i-4}^3$	$h_{i-2}^3, h_{i-4}^3, h_{i+1}^3$	$h_{i+1}^3, h_{i+2}^3, h_{i+3}^3$
E	Int	$h_{i-1}^5, h_{i-2}^5, h_{i-3}^3$	$h_{i-1}^3, h_{i-2}^3, h_{i+1}^3, h_{i+2}^3$	$h_{i+1}^5, h_{i+2}^5, h_{i+3}^3$
	N1	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+1}^5	h_{i+1}^5, h_{i+2}^5
	N2	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+2}^5	h_{i+1}^5, h_{i+2}^5
	C1	$h_{i-1}^3, h_{i-3}^3, h_{i-4}^3$	$h_{i-1}^3, h_{i-3}^3, h_{i+1}^3$	$h_{i+1}^3, h_{i+2}^3, h_{i+3}^3$
	C2	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+1}^5	h_{i+1}^5, h_{i+2}^5
L	Int	$h_{i-1}^5, h_{i-2}^5, h_{i-3}^3, h_{i-4}^3$	$h_{i-1}^5, h_{i-2}^3, h_{i+1}^5, h_{i+2}^3$	$h_{i+1}^5, h_{i+2}^5, h_{i+3}^3, h_{i+4}^3$
	N1	$h_{i-1}^5, h_{i-2}^5, h_{i-3}^3$	$h_{i-1}^5, h_{i-2}^5, h_{i+1}^3$	$h_{i+1}^5, h_{i+2}^5, h_{i+3}^3$
	N2	$h_{i-1}^5, h_{i-2}^3, h_{i-4}^3$	$h_{i-1}^5, h_{i-2}^3, h_{i+1}^3$	$h_{i+1}^5, h_{i+2}^3, h_{i+4}^3$
	N3	$h_{i-1}^5, h_{i-2}^3, h_{i-3}^3$	$h_{i-1}^5, h_{i-2}^3, h_{i+1}^3$	$h_{i+1}^5, h_{i+2}^3, h_{i+3}^3$
	N4	$h_{i-1}^5, h_{i-2}^3, h_{i-3}^3$	$h_{i-1}^5, h_{i-2}^3, h_{i+1}^3$	$h_{i+1}^5, h_{i+2}^3, h_{i+3}^3$
	C1	$h_{i-1}^5, h_{i-2}^5, h_{i-3}^3$	$h_{i-1}^5, h_{i-2}^5, h_{i+1}^3$	$h_{i+1}^5, h_{i+2}^5, h_{i+3}^3$
	C2	h_{i-1}^5, h_{i-2}^5	h_{i-1}^5, h_{i+3}^5	h_{i+1}^5, h_{i+2}^5
	C3	$h_{i-1}^3, h_{i-2}^3, h_{i-3}^3$	$h_{i-1}^3, h_{i+1}^3, h_{i+2}^3$	$h_{i+1}^3, h_{i+2}^3, h_{i+3}^3$
	C4	$h_{i-1}^3, h_{i-2}^3, h_{i-3}^3$	$h_{i-2}^3, h_{i-3}^3, h_{i+1}^3$	$h_{i+1}^3, h_{i+2}^3, h_{i+3}^3$

Table 32: Positional dependencies within structural segments for the models $\mathcal{M}1$, $\mathcal{M}2$, and $\mathcal{M}3$ of the IPSSP-simp method evaluated on the CB513 set. $h_j^3 \in \{hydrophobic, neutral, hydrohilic\}$ indicates the hydrophobicity class of the amino acid R_j, where hydrophobic=$\{A, M, C, F, L, V, I\}$, neutral=$\{P, Y, W, S, T, G\}$, hydrophilic=$\{R, K, N, D, Q, E, H\}$. h_j^5 is a 5 letter alphabet with groups defined as $\{P, G\}$, $\{E, K, R, Q\}$, $\{D, S, N, T, H, C\}$, $\{I, V, W, Y, F\}$, $\{A, L, M\}$.

		$\mathcal{M}1$	$\mathcal{M}2$	$\mathcal{M}3$
H	Int	h_{i-2}^3, h_{i-3}^3, h_{i-4}^3	h_{i-2}^3, h_{i-4}^3, h_{i+2}^3, h_{i+4}^3	h_{i+2}^3, h_{i+3}^3, h_{i+4}^3
	N1	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+2}^3	h_{i+2}^3, h_{i+4}^3
	N2	h_{i-1}^3, h_{i-2}^3	h_{i-2}^3, h_{i+2}^3	h_{i+2}^3, h_{i+4}^3
	N3	h_{i-2}^3, h_{i-3}^3	h_{i-2}^3, h_{i+2}^3	h_{i+1}^3, h_{i+2}^3
	N4	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+2}^3	h_{i+2}^3, h_{i+4}^3
	C1	h_{i-2}^3, h_{i-4}^3	h_{i-2}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	C2	h_{i-2}^3, h_{i-4}^3	h_{i-2}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
E	Int	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3, h_{i+2}^3	h_{i+1}^3, h_{i+2}^3
	N1	h_{i-1}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3
	N2	h_{i-1}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3
	C1	h_{i-1}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3
	C2	h_{i-1}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3
L	Int	h_{i-1}^3, h_{i-2}^3, h_{i-3}^3	h_{i-1}^3, h_{i-2}^3, h_{i+1}^3, h_{i+2}^3	h_{i+1}^3, h_{i+2}^3, h_{i+3}^3
	N1	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	N2	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	N3	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	N4	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	C1	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	C2	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	C3	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3
	C4	h_{i-1}^3, h_{i-2}^3	h_{i-1}^3, h_{i+1}^3	h_{i+1}^3, h_{i+2}^3

3.3 An N-best Approach for Secondary Structure Prediction

The availability of an N-best list enables us to choose from the following options: (1) Combine the set of best scoring M segmentations by a weighted majority voting procedure and arrive at a consensus prediction; (2) Update the score of each segmentation with more sophisticated functions and compute the final prediction as in (1); (3) Keep the suboptimal segmentations so that they can be used by 3-D structure prediction methods or in expert evaluation. The third option can be considered with or without a score update procedure. In this section, we propose the utilization of an N-best list to predict the secondary structure for a given amino acid sequence. To compute the suboptimal segmentations, one can use the modified stack decoder or the N-best Viterbi algorithm, in which the score of a segmentation is defined as the joint probability of the amino acid sequence and the secondary structure labeling, *i.e.*, $P(\mathbf{R}, \mathbf{S}, \mathbf{T})$. Note that while N-best Viterbi generates the Viterbi result (MAP estimation) as the highest scoring segmentation the modified stack decoder might not. Therefore, when the modified stack decoder algorithm is used as the N-best list generator, the most likely state sequence is separately computed by the Viterbi algorithm and is included into the N-best list if it scores higher than the top segmentation in the list. On the other hand, when the N-best Viterbi algorithm is used, the MAP segmentation should be contained in the N-best list and there is no need to compute it separately. Once the N-best list is computed, the segmentations can be re-scored using more elaborate score functions. Then, the final prediction sequence can be computed by applying a weighted voting procedure to a set of best scoring M segmentations. Here, each sequence is weighted by its segmentation score and the same score is applied to all positions within the sequence. The predicted state at position i ($i = 1, .., n$) is computed as the secondary structure type with the highest sum of scores. Setting $M = 1$ reduces to selecting the most likely segmentation as

the prediction sequence. The steps of the method are illustrated in Figure 10.

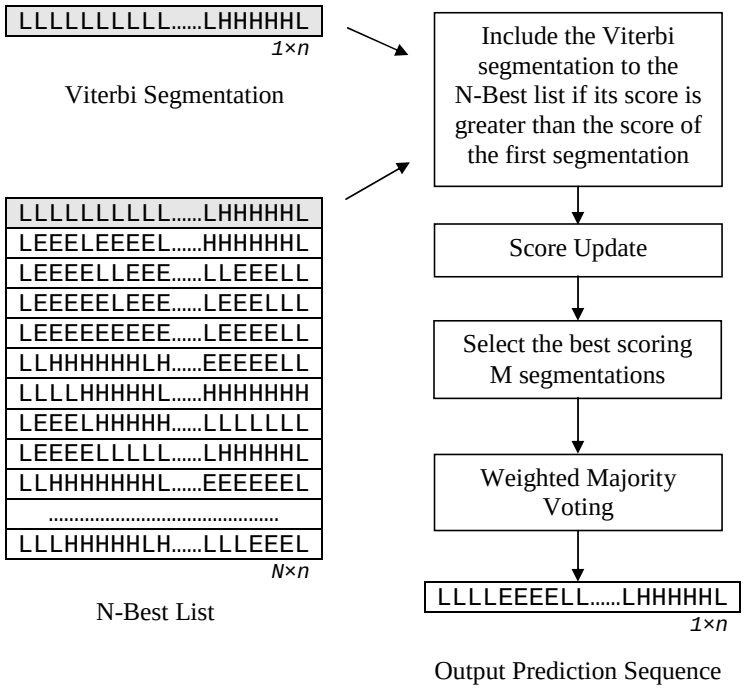


Figure 10: Secondary structure prediction with near-optimal segmentations. N-best Viterbi algorithm does not require the extra computation of the Viterbi (MAP) segmentation and proceeds with the score update after the N-best list generation step.

3.4 Score Update

In this thesis, we utilize the following score distributions to update the score of a segmentation:

3.4.1 Marginal *A Posteriori* Distribution

Let a suboptimal segmentation be denoted as $(\mathbf{S}^{(j)}, \mathbf{T}^{(j)}) = (T_{R_1}^{(j)}, T_{R_2}^{(j)}, ..., T_{R_n}^{(j)})$, where $T_{R_i}^{(j)}$ is the secondary structure type of the i^{th} amino acid in the j^{th} segmentation. Then, the score of this segmentation is updated as $\mathcal{U}^{(j)} = \sum_i P(T_{R_i}^{(j)} \mid R)$, where $P(T_{R_i}^{(j)} \mid R)$ is the *a posteriori* probability of the secondary structure state at position i computed using the forward-backward algorithm (see Section 2.2.4).

3.4.2 Joint Distribution

The joint distribution of the amino acid sequence and a secondary structure segmentation, $P(\mathbf{R}, \mathbf{S}^{(j)}, \mathbf{T}^{(j)})$, is used as the segmentation score and is updated as follows. If $(\mathbf{S}^{(j)}, \mathbf{T}^{(j)})$ contains only α-helices and loops, then $P(\mathbf{R}, \mathbf{S}^{(j)}, \mathbf{T}^{(j)})$ can be computed as in Section 2.2.1. If $(\mathbf{S}^{(j)}, \mathbf{T}^{(j)})$ contains a single β-strand segment, then $P(\mathbf{R}, \mathbf{S}^{(j)}, \mathbf{T}^{(j)})$ is set to zero. If on the other hand, $(\mathbf{S}^{(j)}, \mathbf{T}^{(j)})$ contains two or more β-strands, then the segmentation score is computed using the non-local interaction model explained in the next section.

3.5 A Non-Local Interaction Model for Protein Secondary Structure Prediction

In this section, we develop a Bayesian framework to model the non-local interactions between β-strands. A non-local interaction model has to capture the intrinsic properties of β-sheets, which are explained in Sections 1.1 and 1.2. For a given suboptimal segmentation that contains at least two β-strands, β-sheet groups and interaction types are not defined. Therefore, there can be numerous ways to group β-strands into β-sheets, specify the spatial ordering of β-strands in a β-sheet, and identify the type of interaction between each segment pair. Moreover, due to possible length differences between β-strand segments, there can be many alternatives to align amino acid pairs that make hydrogen bonding contacts. In Fig. 3, two possibilities are shown for the amino acid pairing pattern of a β-sheet that has three β-strand segments.

To include these constraints into the model, we modify the computation of the segmentation score, $P(\mathbf{R}, \mathbf{S}, \mathbf{T})$, as follows[4]. Let $(\mathbf{S}, \mathbf{T})$ contain r β-strand segments $(\mathcal{B}_1, ..., \mathcal{B}_r)$, where $r \geq 2$. We represent the 3-D conformation of these segments by the following components: the grouping of β-strands into β-sheets ($\mathbf{G}$), spatial ordering of β-strands within each β-sheet ($\mathbf{O}$), interaction types of β-strand pairs ($\mathbf{I}$), and

[4]For simplicity, we drop the index j from $(\mathbf{S}^{(j)}, \mathbf{T}^{(j)})$.

the amino acid pairing pattern ($\mathbf{C}$). Detailed definitions of these parameters can be found in Chapter 4. Since the amino acid pairing pattern (or the contact map) contains all the information in the parameter set ($\mathbf{G}, \mathbf{O}, \mathbf{I}$), then the contact map can be used to represent the 3-D conformation assumed by $\mathcal{B}_1, ..., \mathcal{B}_r$. In that case, the score of ($\mathbf{S}, \mathbf{T}$) can be updated as

$$P(\mathbf{R}, \mathbf{S}, \mathbf{T}) = \sum_{\mathbf{C}} P(\mathbf{R}, \mathbf{S}, \mathbf{T}, \mathbf{C}) = \sum_{\mathbf{C}} P(\mathbf{R} \mid \mathbf{S}, \mathbf{T}, \mathbf{C}) P(\mathbf{S}, \mathbf{T}, \mathbf{C}). \tag{37}$$

Using Bayes' rule:

$$P(\mathbf{S}, \mathbf{T}, \mathbf{C}) = P(\mathbf{C} \mid \mathbf{S}, \mathbf{T}) P(\mathbf{S}, \mathbf{T}). \tag{38}$$

In the above equations, $P(\mathbf{S}, \mathbf{T})$ is the *a priori* distribution of ($\mathbf{S}, \mathbf{T}$); $P(\mathbf{C} \mid \mathbf{S}, \mathbf{T})$ is the conditional distribution of the contact map (equivalently the 3-D conformation); and $P(\mathbf{R} \mid \mathbf{S}, \mathbf{T}, \mathbf{C})$ is the sequence likelihood term for a given contact map. $P(\mathbf{S}, \mathbf{T})$ can be computed as in Eq. (3) and $P(\mathbf{C} \mid \mathbf{S}, \mathbf{T})$ is modeled in Chapter 4. In this section, we elaborate on the sequence likelihood term. We start with the following expression:

$$\begin{aligned} P(\mathbf{R} \mid \mathbf{S}, \mathbf{T}, \mathbf{C}) &= \prod_{T_j \in \{H,L\}} P(\mathbf{R}_{[S_{j-1}+1:S_j]} \mid \mathbf{S}, \mathbf{T}) \\ &\times \prod_{k=1}^{w} P(\mathbf{R}_{seg_1(k)}, .., \mathbf{R}_{seg_{n_k}(k)} \mid \mathbf{S}, \mathbf{T}, \mathbf{C}), \end{aligned} \tag{39}$$

where w is the total number of β-sheets in $\mathbf{C}$, such that each sheet contains n_k β-strand segments satisfying $\sum_k n_k = r$ and $\mathbf{R}_{seg_i(k)}$ is the i^{th} β-strand of the k^{th} β-sheet in the spatial order. Here, the spatial ordering of the β-strands is defined by the parameter $\mathbf{O}$, which indexes the β-strand segments in each β-sheet. In this representation, $seg_1(k)$ is the sequential index of the *apriori* β-strand, which is defined as the edge segment with the lowest sequential index[5] in the k^{th} β-sheet. Note that the *a priori* β-strand segment does not have to be the first segment in the sequential

[5] An edge segment makes only one segmental interaction.

ordering. For instance, for a single β-sheet with $\mathbf{O} = (2, 3, 1, 4)$, $\mathbf{R}_{seg_1(k)}$ is the second β-strand segment in the sequential order.

In Eq.(39), the segment likelihoods of α-helices and loops are computed the same as before (see Eq. (5)) and those of β-strands are obtained from the non-local model. The computation of the joint probability term for a β-sheet can be simplified as

$$\begin{aligned} P(\mathbf{R}_{seg_1(k)}, ..., \mathbf{R}_{seg_{n_k}(k)} \mid \mathbf{S}, \mathbf{T}, \mathbf{C}) &= P(\mathbf{R}_{seg_1(k)} \mid \mathbf{S}, \mathbf{T}, \mathbf{C}) \\ &\times \prod_{m=2}^{n_k} P(\mathbf{R}_{seg_m(k)} \mid \mathbf{R}_{seg_{m-1}(k)}, \mathbf{S}, \mathbf{T}, \mathbf{C}). \end{aligned} \tag{40}$$

In this formulation, we assume that a β-strand only depends on its spatial neighbors. We also assume that a β-strand makes at most two segmental interactions. Although this is not always true, we first model the simplest case, in which the β-strands in a β-sheet form a ladder-like topology as shown in Figure 3. The Markovian dependency structure in Eq. (40) can easily be extended for conformations that contain β-strands with more than two segmental partners.

To elaborate further, we should model the terms $P(\mathbf{R}_{seg_1(k)} \mid \mathbf{S}, \mathbf{T}, \mathbf{C})$, and $P(\mathbf{R}_{seg_m(k)} \mid \mathbf{R}_{seg_{m-1}(k)}, \mathbf{S}, \mathbf{T}, \mathbf{C})$ by including the residue interaction propensities in β-strands and other constraints that stabilize the overall structure of β-sheets. The first term, $P(\mathbf{R}_{seg_1(k)} \mid \mathbf{S}, \mathbf{T}, \mathbf{C})$, can be modeled using local dependencies as described in Section 2.2.1. The second term, $P(\mathbf{R}_{seg_m(k)} \mid \mathbf{R}_{seg_{m-1}(k)}, \mathbf{S}, \mathbf{T}, \mathbf{C})$, is the conditional probability of observing amino acids at the remaining segments. Here, the dependency of $\mathbf{R}_{seg_m(k)}$ to $\mathbf{R}_{seg_{m-1}(k)}$ allows us to model dependencies from non-local interactions. We model $P(\mathbf{R}_{seg_m(k)} \mid \mathbf{R}_{seg_{m-1}(k)}\mathbf{S}, \mathbf{T}, \mathbf{C})$ as follows:

$$\begin{aligned} P(\mathbf{R}_{seg_m(k)} \mid \mathbf{R}_{seg_{m-1}(k)}, \mathbf{S}, \mathbf{T}, \mathbf{C}) &= \prod_{u,v} P^{IT}(R^{(u)}_{seg_m(k)} \mid R^{(v)}_{seg_{m-1}(k)}\mathbf{S}, \mathbf{T}, \mathbf{C}) \\ &\times \prod_{u} P^{IT}(R^{(u)}_{seg_m(k)} \mid \mathbf{S}, \mathbf{T}, \mathbf{C}), \end{aligned} \tag{41}$$

where $R^{(u)}_{seg_m(k)}$ and $R^{(v)}_{seg_{m-1}(k)}$ are the u^{th} and v^{th} amino acids of the segments $seg_m(k)$ and $seg_{m-1}(k)$, respectively. The first product term models the amino acid pairs that

interact through non-local contacts as defined by $\mathbf{C}$. The second product is used to model the amino acids of the segment $seg_m(k)$, which do not have a non-local partner in the segment $seg_{m-1}(k)$ because of the length differences. An example to this condition can be found in Figure 3(a), where the amino acid G in the middle segment does not have a partner amino acid with the previous segment (the segment at the top). In Eq. (41), the superscript IT of the probability distributions represents the interaction type of the segments $seg_m(k)$ and $seg_{m-1}(k)$ (parallel or anti-parallel) as defined by the parameter $\mathbf{I}$. For instance, for the β-sheets in Figure 3, $\mathbf{I} = (AP, AP)$, where AP denotes an anti-parallel interaction. The probability distributions on the right hand side of Eq. (41) can easily be estimated using the frequency of occurrence counts in the available training data. This completes the derivation of the non-local interaction model for protein secondary structure prediction.

3.6 Results

In our simulations, we evaluated the performance on the EVA and CB513 sets (see Section 2.5.1.1). We removed sequences that contained secondary structure segments longer than $D = 40$ amino acids similar to Chapter 2. In our simulations with the N-best algorithms, we removed proteins shorter than 30 to further refine the datasets. We also removed proteins longer than 400 amino acids to prevent our evaluations slow down due to long proteins. After applying these constraints, there remained 2251 proteins in the EVA set, and 447 proteins in the CB513 set. The eight state secondary structure assignments for the proteins in the datasets were taken from the PDB database[6]. To reduce the eight secondary structure state assignment used in the DSSP notation to three, we used the following conversion rule: H to H; E to E; all other states to L, which is also known as the 'CK' mapping [59, 49]. We also considered using the length adjustments proposed by Frishman and Argos [58] that

[6]PDB uses the DSSP algorithm for the assignment of the secondary structure from the atomic coordinates.

convert the α-helices shorter than five amino acids and β-strands shorter than three amino acids to loops.

In all simulations, we performed a leave-one-out cross-validation experiment. For parameter estimation, we used the maximum-likelihood estimation procedure where we count the observed frequencies for the desired quantities and apply a proper normalization factor to compute the probabilities. To evaluate the performance, we chose the three-state-per-residue accuracy, Q_3, and the Segment OVerlap score, SOV, as the overall accuracy measures. Detailed descriptions of the leave-one-out cross validation, the maximum-likelihood estimation and the accuracy measures can be found in Chapter 2.

3.6.1 N-best Predictors without Score Update

In this section, we compare the performances of the Viterbi algorithm, the N-best algorithms, and the IPSSP method. In the first set of simulations, we did not apply any score update to the N-best list. Then, we evaluated the effect of updating the segmentation scores with the marginal *a posteriori* distribution $P(T_{R_i} \mid R)$ obtained by the IPSSP method and with the non-local interaction model. The dependency patterns (feature sets) employed by the methods evaluated is described in Section 3.2.3. To initialize the frequency tables, Laplace's rule is used as the pseudo-count method, in which the entries are initially set to one.

3.6.1.1 Modified Stack Decoder vs N-best Viterbi

We first compared the performances of the Viterbi, the modified stack decoder, and the N-best Viterbi algorithms. We chose the size of the N-best list as $N = 30,000$. In this set of simulations, we did not apply any score update. To obtain the final prediction sequence, we combined the best scoring $M = 5000$ segmentations by the weighted voting procedure as explained in Section 3.3. We chose K as three for the N-best Viterbi algorithm. From Table 33, the modified stack decoder algorithm performs

better than the Viterbi algorithm by 1.1% in terms of the Q_3 measure. For the N-best Viterbi algorithm, the improvement is only 0.25% because the N-best Viterbi generates a significantly higher number of sequences with scores close to the most likely sequence. In addition, the score differences are smaller for the N-best Viterbi algorithm. The overall accuracy of the N-best Viterbi can be improved by increasing the size of the N-best list. A comparison of the structure-type-specific measures shows that the N-best Viterbi algorithm has the highest Q_α^{obs} and Q_β^{obs} values, followed by the Viterbi algorithm. The highest loop sensitivity Q_L^{obs} is achieved by the modified stack decoder algorithm. These results show that the information in suboptimal segmentations is useful and is capable of improving the MAP estimation even when there is no score update.

Table 33: Sensitivity results of the Viterbi, modified stack decoder and N-best Viterbi algorithms. In simulations with the N-best algorithms weighted majority voting is applied to a set of top scoring M segmentations.

Sensitivity	$Q_3(\%)$	$Q_\alpha^{obs}(\%)$	$Q_\beta^{obs}(\%)$	$Q_L^{obs}(\%)$
Viterbi	64.17	64.99	28.70	76.56
Modified Stack Decoder	65.28	60.42	26.78	79.95
N-best Viterbi	64.42	65.41	28.74	76.77

3.6.1.2 N-best List Size

In this section, we investigate the effect of changing the N-best list size N, and the number of voting sequences M. Table 34 shows the sensitivity results of the proposed method for different values of N and M. Here, the suboptimal segmentations are obtained using the N-best Viterbi algorithm with $K = 3$. From Table 34, increasing the size of the N-best list improved the Q_3, Q_α^{obs} and Q_L^{obs} measures. For a given list size, increasing the number of voting sequences improved only the Q_L^{obs} measure. The results demonstrate that suboptimal segmentations contain valuable information and

can improve the accuracy when the list size is increased (a deeper list) and the segmentations are sampled more densely (with scores close to each other). The decrease in the β-strand sensitivity for increasing values of N can be explained by the fact that the current statistical model can only capture local interactions, which are dominantly observed in α-helices and loops. Therefore, without incorporating additional knowledge sources, N-best methods are not expected to improve the accuracy of the β-strand predictions.

Table 34: Sensitivity results of the N-best Viterbi algorithm for changing values of N and M.

N	M	$Q_3(\%)$	$Q_\alpha^{obs}(\%)$	$Q_\beta^{obs}(\%)$	$Q_L^{obs}(\%)$
30,000	500	64.422	65.413	28.750	76.775
30,000	5,000	64.421	65.411	28.748	76.775
50,000	5,000	64.446	65.514	28.559	76.833
100,000	10,000	64.519	65.526	28.544	76.974

3.6.2 N-best Predictors with Score Update

3.6.2.1 Score Update with Marginal A Posteriori Distribution

In this section, we investigate the effect of updating the segmentation scores using the *a posteriori* probability distribution $P(T_{R_i} \mid R)$ as described in Section 3.3. We compare the performances of the Viterbi algorithm, the IPSSP algorithm, and the N-best algorithms. To compute suboptimal segmentations, we used the N-best Viterbi algorithm with $N = 1,000,000$ and $K = 4$. For the number of voting sequences used in the weighted majority voting step, we chose two different values: $M = 1$ and $M = 10,000$. The results of the cross-validation experiments are shown in Tables 35 and 36 for the EVA set and in Tables 37 and 38 for the CB513 set. In EVA set simulations, we used the IPSSP algorithm, which takes the ensemble average of three dependency models, each calibrated by a training set reduction procedure (see

Chapter 2 and [23]). In simulations with the refined CB513 set, we used the IPSSP-simp method, which employs reduced versions of the IPSSP's dependency models (see Section 3.2.3). In both versions of IPSSP, the threshold used in the training set reduction step is set to 0.35. For all methods, the Laplacian pseudo-count method is applied to initialize the frequency tables before estimating the model parameters. The $P(T_{R_i} \mid R)$ values that are used to obtain the IPSSP predictions and to update segmentation scores are computed as described in Chapter 2 and in Aydin *et al.* [23], in which the posterior probability distributions from three dependency models are averaged (see Section 3.2.3).

Table 35: Sensitivity results of the Viterbi, IPSSP, and N-best Viterbi with score update, evaluated on the reduced EVA set under leave-one-out cross-validation.

Sensitivity	$Q_3(\%)$	$Q_\alpha^{obs}(\%)$	$Q_\beta^{obs}(\%)$	$Q_L^{obs}(\%)$
Viterbi	63.95	65.66	24.37	77.30
IPSSP	70.06	66.77	45.25	80.93
N-best Viterbi Score Update (M = 1)	66.52	65.43	30.83	80.08
N-best Viterbi Score Update (M = 10,000)	65.80	65.67	29.06	79.18

Table 36: SOV measures of the Viterbi, IPSSP, and N-best Viterbi with score update, evaluated on the reduced EVA set under leave-one-out cross-validation.

SOV Score	$SOV_3(\%)$	$SOV_\alpha(\%)$	$SOV_\beta(\%)$	$SOV_L(\%)$
Viterbi	51.45	64.47	27.63	52.57
IPSSP	64.09	70.99	54.63	63.55
N-best Viterbi Score Update (M = 1)	54.74	66.32	35.48	55.06
N-best Viterbi Score Update (M = 10,000)	53.55	65.93	33.54	53.67

Table 37: Sensitivity results of the Viterbi, IPSSP-simp, and N-best Viterbi with score update, evaluated on the reduced CB513 set under leave-one-out cross-validation.

Sensitivity	$Q_3(\%)$	$Q_\alpha^{obs}(\%)$	$Q_\beta^{obs}(\%)$	$Q_L^{obs}(\%)$
Viterbi	61.93	69.44	28.04	73.05
IPSSP-simp	67.91	68.38	48.80	76.65
N-best Viterbi Score Update (M = 1)	64.69	69.17	36.67	75.01
N-best Viterbi Score Update (M = 10,000)	63.71	70.10	33.88	73.64

Table 38: SOV measures of the Viterbi, IPSSP-simp, and N-best Viterbi with score update, evaluated on the reduced CB513 set under leave-one-out cross-validation.

SOV Score	$SOV_3(\%)$	$SOV_\alpha(\%)$	$SOV_\beta(\%)$	$SOV_L(\%)$
Viterbi	52.34	67.50	31.52	52.27
IPSSP-simp	64.39	70.78	56.44	63.98
N-best Viterbi Score Update (M = 1)	56.00	67.91	41.44	55.09
N-best Viterbi Score Update (M = 10,000)	54.47	67.72	38.20	53.51

From these results, the score update procedure yields an average improvement of 2.6% over the Viterbi algorithm in terms of the three-state-per-residue accuracy $Q_3(\%)$ (Tables 35 and 36) and an average improvement of 3.5% in terms of the SOV measure as (Tables 37 and 38). In both simulations, choosing the most likely segmentation as the prediction performed better than the consensus approach (weighted voting) on a set of most likely segmentations. This result can be explained by the fact that, after applying a score update, the first M segmentations get more diverse for increasing values of M, and hence less accurate ones are likely to be selected. Hence, when a score update is performed, it is better to choose the most likely segmentation as the final prediction sequence.

3.6.2.2 Score Update with Non-Local Interaction Model

In this set of simulations, we performed a leave-one-out cross validation experiment on the PDB_SELECT dataset (see Section 2.5.1.1). We removed proteins shorter than 30 amino acids and obtained a representative set of 2482 sequences. To generate suboptimal segmentations (*i.e.*, N-best list), we used the N-best Viterbi algorithm introduced in Section 3.2.2 and in Aydin *et al.* [24]. Here, we considered two scenarios: (1) true secondary structure segmentation is manually included into the N-best list; (2) true secondary structure segmentation is not included into the N-best list. We also tested whether incorporating the number of β-strands as an *a priori* information will improve the results. This is achieved by setting the scores of the secondary structure segmentations that do not have exactly the same number of β-strands as the true segmentation to zero. We evaluated the performance of our method on all-β proteins[7] of the PDB_SELECT dataset with ≤ 3 β-strands. There were 49 proteins in PDB_SELECT that satisfy these conditions.

The results of the cross validation experiment are shown in Table 39. In the Viterbi algorithm implementation, we used the all-β information in computing the secondary structure prediction. For the N-best method, we utilized both the all-β information and the information on the number of β-strands. From Table 39, although the β-strand accuracy of the N-best method improved, the loop accuracy and hence the overall accuracy decreased. Considering that the marginal posterior mode algorithm (posterior decoding) performs 3-4% better than the Viterbi algorithm in terms of the overall sensitivity, and approximately 15-20% better in terms of the β-strand sensitivity, the performance of the N-best method is not satisfactory to be regarded as a significant improvement.

[7]Proteins with β-strands and loops only.

Table 39: Sensitivity results of the Viterbi and the N-best method with the non-local model, evaluated on the reduced PDB_SELECT set under the leave-one-out cross-validation. The Viterbi algorithm uses all-β information in computing the secondary structure prediction. The N-best Method uses both all-β and the number of β-strands information. In the third row, true secondary structure segmentation is included into the N-best list. The N-best list parameters are chosen as N = 1,000,000, $M = 1$.

Sensitivity	$Q_3(\%)$	$Q_\alpha^{obs}(\%)$	$Q_\beta^{obs}(\%)$	$Q_L^{obs}(\%)$
Viterbi	76.455	—	26.774	88.097
N-best Viterbi with NL Model (true ss not inc.)	73.241	—	34.783	82.252
N-best Viterbi with NL Model (true ss inc.)	74.717	—	37.986	83.324

In addition to the sensitivity evaluation, we analyzed the rank of the true segmentation after updating the segmentation scores using the non-local interaction model. We have found that when the test protein had only two β-strands, the rank of the true segmentation was less than 20. However, for proteins with 3 β-strands, the ranks increased up to 500.

These results show that a non-local interaction model based on the amino acid residue-residue interaction propensities is not sufficient to discriminate the true segmentation from the incorrect ones, even if the all-β information and the information on the number of β-strand segments are incorporated. This is mainly due to the uninformative behavior of the amino acid residue-residue interaction propensities and

is consistent with the earlier findings. Cline *et al.* [102] and Crooks *et al.* [61] examined the mutual information content of the interacting amino acid residues distantly separated by sequence but proximate in 3-D structure and concluded that for the purposes of tertiary structure prediction, these interactions are essentially uninformative. Therefore, in the single-sequence setting, improving the prediction accuracy by incorporating non-local interactions is impractical.

3.7 Summary

In this chapter, we developed two N-best algorithms and a non-local interaction model for protein secondary structure prediction in the single-sequence setting. We showed that the information in suboptimal segmentations is useful and can improve the sensitivity of the Viterbi algorithm up to 1% without applying any score update. When the segmentations are re-scored using the marginal posterior probability distribution, the improvement becomes 2.6%. Unfortunately, the two N-best algorithms and the score update procedure were not able to perform better than the posterior decoding algorithm in the single-sequence predictions. Also, the incorporation of non-local interactions did not leverage the accuracy due to uninformative behavior of amino acid interaction potentials in the single-sequence condition.

CHAPTER IV

PROTEIN BETA-SHEET PREDICTION WITH BAYESIAN MODELS AND ALGORITHMS

4.1 Introduction

Several methods have been proposed for the prediction of β-strand organizations and contact maps (see Section 1.3.2). Although the BetaPro method proposed by Cheng and Baldi [39] is one of the best methods developed to date, it has limitations. First of all, BetaPro does not explicitly employ folding rules and does not discriminate between possible topological organizations. In other words, it treats possible groupings of β-strands into β-sheets, spatial ordering of β-strands within a sheet and interaction types of β-strand pairs equally. In a related study, Ruczinski *et al.* [131] showed that the organization of β-strands into β-sheets is not random and shows a distinct pattern. Some of the conformations are physically unstable and are never observed. For the remaining ones, there is a preference for particular orientations, which are favored more than the others. Moreover, although Cheng and Baldi [39] defined and introduced a gapped alignment algorithm for β-strand interactions, they did not implement gapped alignments in BetaPro. They simply ignored gaps by sliding one segment along with the other. Another aspect of BetaPro is that it employs a simple greedy algorithm to compute β strand pairings and interaction types. This leaves room for more sophisticated algorithms to be developed.

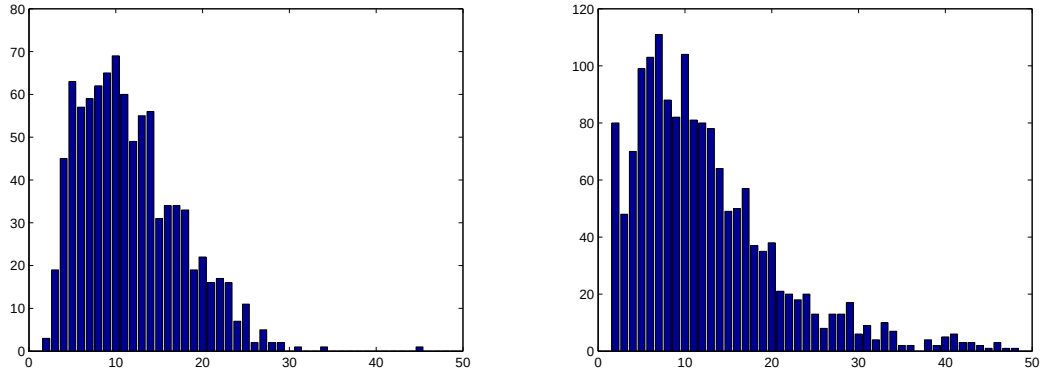
To improve BetaPro, Jeong et al. [79] investigated two new algorithms for predicting β-strand partners. To make direct comparisons, they used the same scoring function as of BetaPro. The objective of the first algorithm is that instead of having

a two-stage greedy selection heuristic, it poses the problem as integer linear programming optimization problem and solves it using the ILOG CPLEX^{TM} package. The second approach is greedy and it explicitly encourages two simple folding rules. This is achieved by dynamically increasing the scores of strand pairs that are potential partners depending on the pairs predicted so far. The second algorithm performed better than the first one but the improvement over BetaPro was not drastic (a 0.7% improvement in sensitivity and 2.7% improvement in positive predictive value evaluated in the β-strand pairing category). Also they did not report the accuracy in interaction type and contact map predictions. Furthermore, their accuracy was not better than BetaPro for all separation distances between contacts. For some distances the accuracy decreased slightly. In both algorithms, Jeong et al. [79] extended the dynamic programming approach that computes pairwise alignments of β-strands. However, they allowed only a single gap in an alignment. More importantly, although Jeong *et al.* [79] aimed to enforce physical constraints by incorporating folding rules into BetaPro, they considered only two simple folding rules. Therefore, for an elaborate treatment of the problem, one has to include a more comprehensive set of rules and physical preferences that guide the formation of β-sheets.

Recently, BetaPro was followed by SVMcon, a new contact map predictor that uses support vector machines to predict medium- and long-range contacts [40]. Although SVMcon utilized a larger feature set, its performance was not better than BetaPro when evaluated on CASP datasets [40].

In this chapter, we address the problem of β-sheet prediction defined as the prediction of β-strand pairings, interaction types (parallel or anti-parallel), and β-residue interactions (or contact maps). We analyze proteins according to the number of β-strands they contain. We consider two categories: (1) proteins with six or less β-strands; (2) proteins with more than six β-strands. In Figure 11, histogram plots are shown for the number of β-strands in BetaSheet916 and CulledPDB datasets (see

Section 4.2.3). The percentage of proteins with six or less β-strands is calculated as 20.41% in BetaSheet916 and 25.51% in CulledPDB. Due to limitations in the avail-



(a) BetaSheet916 (b) CulledPDB

Figure 11: Histograms for the number of β-strands in an amino acid chain

able training data, we mainly target proteins in the first category. We introduce a Bayesian approach for proteins with six or less β-strands, in which we model the conformational features in a probabilistic framework by combining the amino acid pairing potentials with *a priori* knowledge of β-strand organizations. Starting from the amino acid sequence, secondary structure, and the amino acid pairing probability matrix computed by BetaPro, we assign probability scores to possible β-sheet architectures by considering four structure levels: (1) groupings of β-strands into β-sheets; (2) spatial arrangement of β-strands in each β-sheet; (3) interaction types of β-strands (parallel or anti-parallel); (4) residue pairing patterns (or contact maps). For the first three levels, we utilize the results of Ruczinski *et al.* [131], who performed a statistical analysis of the frequency of β-strand groupings and β-sheet motifs. For the fourth level, we use the raw amino acid pairing probabilities that are derived from the DSSP database [11, 81][1]. This approach allows us to enforce a large set of physical rules that characterize the intrinsic preferences of β-sheet formation.

[1]The BetaPro's pairing probability matrix is not used in scoring the conformations

To select the optimum β-sheet architecture, we search the space of possible conformations by efficient heuristics. In our computations, we significantly reduce the search space by enforcing the amino acid pairs with strong interaction propensities derived from the residue pairing propensity matrix. On this reduced search space, we sample the first three levels using a brute-force sampling approach. To derive the optimum amino acid pairing combination (*i.e.*, the contact map), we apply dynamic programming and compute pairwise alignments of β-strand pairs. For this purpose, we employ an algorithm that finds the optimum pairwise alignment of β-strands. In this algorithm, we define match as well as gap scores and perform global alignments (Needleman-Wunsch algorithm). This is a more elaborate approach as compared to the earlier work by Cheng and Baldi [39] and Jeong *et al.* [79]. We further improved the dynamic programming approach by allowing any number of gaps. The gapped nature of the alignments enables us to model β-bulges more effectively.

For proteins with more than six β-strands, the discriminative power of the Ruczinski model reduces significantly due to an exponential increase in the number of possible β-strand organizations. Therefore, for such proteins, we first use BetaPro to compute β-strand pairings. Then, we compute gapped alignments of the paired β-strands in parallel and anti-parallel directions and choose the interaction types and the β-residue pairing patterns with maximum alignment scores.

4.2 Methods

4.2.1 Beta-Sheet Prediction for Proteins with $\leq$ 6 Beta-Strands: A Bayesian Approach

We will formulate the β-sheet prediction in a probabilistic framework. Before providing the mathematical details, we first define our model parameters.

4.2.1.1 Model Parameters

The input parameters are the amino acid sequence, the secondary structure, and an amino acid pairing propensity matrix. The amino acid sequence is denoted by $\mathbf{R}$, where $\mathbf{R}_i$ is the i^{th} amino acid. Similarly, the secondary structure is represented by $\mathbf{SS}$, where $\mathbf{SS}_i$ is the secondary structure state of the i^{th} amino acid (whether it is α-helix, β-strand or loop). Note that as a necessary condition for β-sheet formation, $\mathbf{SS}$ should contain at least two β-strand segments. The third parameter is denoted by $\mathbf{PP}$, where $\mathbf{PP}_{ij}$ is the probability of the i^{th} and j^{th} β-residues to make a pair (or contact). In this matrix, only the amino acids in β-strands or β-bridges are considered (*i.e.*, E or B states in the DSSP assignment [81]). The pairing probability matrix is computed using the BetaPro method [39] and is utilized to reduce the space of possible β-sheet conformations.

The output parameters are the grouping sequence $\mathbf{G}$, the ordering sequence $\mathbf{O}$, the interaction type sequence $\mathbf{I}$, and the contact map (or the residue pairing sequence) $\mathbf{C}$. We explain each parameter in more detail.

- $\mathbf{G}$ defines the number of β-sheets as well as the grouping of β-strands into β-sheets. In other words, $\mathbf{G}$ contains the information about which β-strands appear together in each β-sheet. Here, the ordering of β-strands is not important, therefore they are ordered in the sequential order to remove ambiguity. $\mathbf{G}$ is a 2D sequence, where $\mathbf{G}(p, l)$ is the sequence index of the l^{th} β-strand in the p^{th} β-sheet. For the β-sheet in Figure 2(a), $\mathbf{G} = (1, 2, 3, 4)$ meaning that all β-strands form a single β-sheet.

- $\mathbf{O}$ specifies the spatial ordering of β-strands within each β-sheet. $\mathbf{O}$ is a 2D sequence, where $\mathbf{O}(p, l)$ is the spatial order of the l^{th} β-strand in the p^{th} β-sheet. If the p^{th} β-sheet contains n_p β-strands, then $\mathbf{O}(p, :)$ (also denoted by $\mathbf{O}_p$) is simply a permutation of the sequence $1{:}n_p$. Therefore, in this notation, $\mathbf{O}$ can

be represented as the concatenation of $\mathbf{O}_p$'s. This is formulated as $\mathbf{O} = \Upsilon_p \mathbf{O}_p$, where Υ is the sequence concatenation operator. Note that in our model, a permutation and its inverse represent the same spatial ordering because we only keep permutations, in which the sequential index of the first segment is lower than the index of the last segment. The spatial ordering information also specifies the β-strand segments that interact with each other. For the β-sheet in Figure 2(a), $\mathbf{O} = (1, 2, 4, 3)$ meaning that the first β-strand interacts with the second, the second with the fourth, and the fourth with the third. The pairwise interactions are bidirectional. Here, for simplicity, we assume that a segment can interact with up to two neighboring segments. The percentage of proteins that have six or less β-strands and that contain interactions with more than two segments is only 1.7% in the BetaSheet916 set (see Section 4.2.3.2). Extension of the model to characterize interactions with more than two neighbors is not a difficult task and is left as a future work (see Section 4.4). Note that for proteins with more than six β-strands, we are not putting any restriction on the number of interactions a β-strand makes (see Section 4.2.2).

- $\mathbf{I}$ determines the interaction types (parallel or anti-parallel) of β-strand pairs in each sheet. $\mathbf{I}$ is a 2D sequence, where $\mathbf{I}(p, l)$ is the interaction type between the l^{th} and $(l+1)^{th}$ β-strands in the p^{th} β-sheet represented in the spatial order. We set $\mathbf{I}(p, l) = P$ if the l^{th} β-strand is parallel to the $(l+1)^{th}$ β-strand. If two neighboring β-strands are anti-parallel, we set $\mathbf{I}(p, l) = AP$. For the β-sheet in Figure 2(a), $\mathbf{I} = (AP, AP, P)$. Similar to the ordering sequence, $\mathbf{I}$ can be decomposed into its subcomponents denoted by $\mathbf{I}_p = \mathbf{I}(p, :)$. This is formulated as $\mathbf{I} = \Upsilon_p \mathbf{I}_p$, where Υ is the sequence concatenation operator.

- $\mathbf{C}$ describes the non-local residue pairing pattern or the contact map arising from the amino acid interactions in each β-sheet. In our model, we assume that

an amino acid can make a residue pairing interaction with up to 2 other amino acids. There can be various formats to represent the contact map. The first one is the classical representation, where a 2D sequence $\bar{\mathbf{C}}$ of size $n_R \times n_R$ is used. Here, n_R is the total number of amino acids labeled as β-strand and the amino acid residues in β-strand segments (β-residues) are indexed following the sequential order (*i.e.*, from the N-terminus to the C-terminus of the protein). $\bar{\mathbf{C}}(i,j)$ is set to 1 if the i^{th} β-residue interacts with the j^{th} β-residue. If there is no interaction between the residue pair, then $\bar{\mathbf{C}}(i,j)$ is set to 0. As an alternative representation, we can only keep the indices of the residue pairs that make residue pairing interaction and store them in a 2D sequence denoted by $\mathbf{C}$. In other words, we only keep the residue indices for which $\bar{\mathbf{C}}$ is 1. In this representation, each row of $\mathbf{C}$ corresponds to a β-sheet and contains the indices of the amino acid residue pairs that make chemical interactions. For instance, if the 5^{th} amino acid interacts with the 3^{rd} and 21^{th} amino acids, and if they all belong to the p^{th} β-sheet then $\mathbf{C}(p,:) = \mathbf{C}_p$ contains (3,5,5,21). This notation is equivalent to the classical contact map representation in the sense that given the secondary structure segmentation $\mathbf{SS}$, it is possible to convert one to the other. Similar to $\mathbf{O}$ and $\mathbf{I}$, $\mathbf{C}$ can be decomposed into subcomponents denoted by $\mathbf{C}_p$. In addition, we can decompose each $\mathbf{C}_p$ into its subcomponents designated by $\mathbf{C}_p^m$. Here, $\mathbf{C}_p^m$ contains the set of residue pairs that connect a pair of β-strands and m runs from 1 to $n_S^p - 1$, where n_S^p is the number of β-strand segments in the p^{th} β-sheet. In that case, $\mathbf{C}_p^m$ can be concatenated to form $\mathbf{C}_p$ and likewise $\mathbf{C}_p$ can be concatenated to form $\mathbf{C}$. This is expressed as $\mathbf{C}_p = \Upsilon_m \mathbf{C}_p^m$ and $\mathbf{C} = \Upsilon_p \mathbf{C}_p$, where Υ is the sequence concatenation operator.

4.2.1.2 Problem Definition

In β-sheet prediction, the goal is to predict the overall β-sheet conformation of the protein given the input variables. Since the contact map $\mathbf{C}$ contains all the information in the parameter set $(\mathbf{G}, \mathbf{O}, \mathbf{I})$, the problem reduces to finding the optimum contact map or the residue pairing structure. This is formulated as

$$\mathbf{C}^{max} = \arg\max_{\mathbf{C}} P(\mathbf{C} \mid \mathbf{D}), \tag{42}$$

where $\mathbf{C}^{max}$ is the MAP estimator, which corresponds to the contact map (or equivalently the conformation) maximizing the *a posteriori* probability $P(\mathbf{C} \mid \mathbf{D})$, and $\mathbf{D}$ is a short-hand notation for $(\mathbf{R}, \mathbf{SS}, \mathbf{PP})$, *i.e.*, the set of input variables defined in Section 4.2.1.1. The posterior probability can be modeled as

$$\begin{aligned} P(\mathbf{C} \mid \mathbf{D}) &= P(\mathbf{C}, \mathbf{G}, \mathbf{O}, \mathbf{I} \mid \mathbf{D}) && (43)\\ &= P(\mathbf{G}, \mathbf{O}, \mathbf{I} \mid \mathbf{D}) P(\mathbf{C} \mid \mathbf{G}, \mathbf{O}, \mathbf{I}, \mathbf{D}) && (44)\\ &= P(\mathbf{G} \mid \mathbf{D}) P(\mathbf{O}, \mathbf{I} \mid \mathbf{G}, \mathbf{D}) P(\mathbf{C} \mid \mathbf{G}, \mathbf{O}, \mathbf{I}, \mathbf{D}) && (45) \end{aligned}$$

Given the grouping vector, which specifies the assignment of β-strands into β-sheets, we model the terms $P(\mathbf{O}, \mathbf{I} \mid \mathbf{G}, \mathbf{D})$ and $P(\mathbf{C} \mid \mathbf{G}, \mathbf{O}, \mathbf{I}, \mathbf{D})$ as

$$\begin{aligned} P(\mathbf{O}, \mathbf{I} \mid \mathbf{G}, \mathbf{D}) &= \prod_k P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{G}, \mathbf{D}), && (46)\\ P(\mathbf{C} \mid \mathbf{G}, \mathbf{O}, \mathbf{I}, \mathbf{D}) &= \prod_k P(\mathbf{C}_k \mid \mathbf{G}, \mathbf{O}_k, \mathbf{I}_k, \mathbf{D}), && (47) \end{aligned}$$

where the vectors $\mathbf{O}_k$, $\mathbf{I}_k$, and $\mathbf{C}_k$ denote the ordering, interaction type and the contact map of the k^{th} β-sheet, respectively. With this formulation, we assume that β-sheets[2] are independent from each other. We further assume that

$$\begin{aligned} P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{G}, \mathbf{D}) &= P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{D}), && (48)\\ P(\mathbf{C}_k \mid \mathbf{G}, \mathbf{O}_k, \mathbf{I}_k, \mathbf{D}) &= P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D}), && (49) \end{aligned}$$

[2]Note that β-strands are not assumed to be independent.

where the arrangements within a β-sheet is modeled as independent from the grouping vector $\mathbf{G}$.

4.2.1.3 Bayesian Models

In this section, we concentrate on the modeling of $P(\mathbf{G} \mid \mathbf{D})$, $P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{D})$, and $P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D})$.

Modeling of $P(\mathbf{G} \mid \mathbf{D})$

We model the grouping term as in Ruczinski [130]:

$$P(\mathbf{G} \mid \mathbf{D}) = P(SD \mid n_{SH}, n_S) P(n_{SH} \mid n_S), \tag{50}$$

where n_S is the number of β-strand segments in $\mathbf{SS}$, n_{SH} is the number of β-sheets in $\mathbf{G}$, SD is the sheet decomposition term, which defines the assignment of β-strands into β-sheets. Analyzing the available data, Ruczinski [130] derived probability models for computing $P(SD \mid n_{SH}, n_S)$ and $P(n_{SH} \mid n_S)$ (see the thesis chapter in [130] or Appendix A.1 for further details). In this thesis, we used the same model as in Ruczinski [130] for the grouping term.

Modeling of $P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{D})$

The vector $(\mathbf{O}_k, \mathbf{I}_k)$ defines a structural unit known as a β-sheet motif. Ruczinski *et al.* [131] surveyed the distribution of β-sheet motifs with two edge strands (open sheets) in a large set of non-homologous proteins (see Section 4.2.3.1). They investigated to what extent the distribution can be accounted for by the rules previously published in the literature. For instance, analyzing the motifs in four-stranded-β-sheets, they have found that 48 out of 96 possible motifs were never observed (a subset of of those motifs were shown in Figure 12). The non-existence of some motifs can be explained by the "absolute" rules for physically impossible β-sheet configurations. In addition to the "absolute" rules, there are also "probabilistic" rules that favor some motifs more than the others. These can be categorized into two major groups: preference for purely parallel and purely anti-parallel β-sheets, and preference

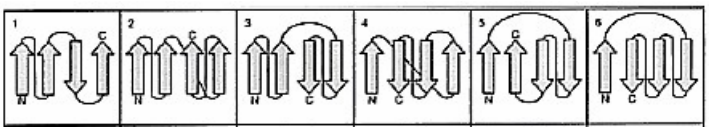


Figure 12: A subset of four-stranded motifs that did not occur in the CulledPDB dataset as evaluated by Ruczinski *et al.* [131].

for maintaining the sequential ordering of β-strands in the spatial ordering. Ruczinski *et al.* [131] also reported that the position of the first β-strand segment in a motif is not random. Based on these findings they developed probabilistic models to compute the motif-likelihood distribution $P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{D})$. We can start with the following simplifying assumption:

$$P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{D}) = P(\mathbf{O}_k, \mathbf{I}_k \mid H, L), \tag{51}$$

where H is the helical status of the protein (helical or non-helical), and L is the connector lengths between the strands given as indicators (short or long). Here, a protein is considered to be helical if at least 20% of its amino acids are part of an α-helix, and a connector is defined as a set of segmental residues, which connect two β-strands. Note that connectors can include α-helices and loops.

Based on the available data, estimation of $P(\mathbf{O}_k, \mathbf{I}_k \mid H, L)$ for proteins with four or less β-strands can be performed using the raw counts from the CulledPDB dataset as shown in Ruczinski *et al.* [131]. For proteins with five or more β-strands, this is not feasible and one has to simplify the probability model. For this purpose, Ruczinski *et al.* [130] identified a representative set of features that characterize $(\mathcal{O}_k, \mathcal{I}_k)$. This approach allows us to model $P(\mathcal{O}_k, \mathcal{I}_k \mid SS_c)$ as

$$P(\mathcal{O}_k, \mathcal{I}_k \mid SS_c) = \frac{P(\mathcal{F}_k \mid SS_c)}{k(\mathcal{F}_k, SS_c)}, \tag{52}$$

where Λ_k is the feature set that represents the motif $(\mathcal{O}_k, \mathcal{I}_k)$ and $k(\mathcal{F}_k, SS_c)$ is the number of motifs that satisfy the constraints in SS_c, which can be estimated from the PDB database (see the thesis chapter of Ruczinski [130] or Appendix A.2 for further

details). In this thesis, we used the same model as in Ruczinski [130] for the motif distribution term.

Modeling of $P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D})$

Let the k^{th} β-sheet contain r β-strand segments, which are represented by $\mathcal{B}_1$, $\mathcal{B}_2$,...,$\mathcal{B}_r$ in the spatial order. We estimate $P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D})$ as

$$P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D}) = \prod_{m=1}^{r-1} P(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D}), \tag{53}$$

where $\mathbf{C}_k^m$ is the residue pairing pattern of the m^{th} segment pair $(\mathcal{B}_m, \mathcal{B}_{m+1})$ of the k^{th} β-sheet. In other words, it is a subset of $\mathbf{C}_k$ and defines the interactions (or contacts) between $\mathcal{B}_m$ and $\mathcal{B}_{m+1}$. Concatenation of $\mathbf{C}_k^m$ with respect to m gives $\mathbf{C}_k$ (see the definition of $\mathbf{C}$ in Section 4.2.1.1). The term $P(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D})$ is computed as the product of the individual amino acid residue pairing probabilities, which is formulated as

$$\begin{aligned} P(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D}) &= \prod_p P(RP = 1 \mid R_m^p, R_{m+1}^p, \mathbf{O}_k, \mathbf{I}_k) \times \prod_q P(RP = 0 \mid R_m^q, \mathbf{O}_k, \mathbf{I}_k) \\ &\times \prod_r P(RP = 0 \mid R_{m+1}^r, \mathbf{O}_k, \mathbf{I}_k), \end{aligned} \tag{54}$$

where $P(RP = 1 \mid R_m^p, R_{m+1}^p, \mathbf{O}_k, \mathbf{I}_k)$ is the probability of the amino acid R_m^p in segment $\mathcal{B}_m$ to interact (or make a contact) with the amino acid R_{m+1}^p in segment $\mathcal{B}_{m+1}$, $P(RP = 0 \mid R_m^q)$ is the probability of the amino acid R_m^q not to make a residue pairing interaction with any amino acid in segment $\mathcal{B}_{m+1}$, and $P(RP = 0 \mid R_{m+1}^r)$ is the probability of the amino acid R_{m+1}^r not to make a residue pairing interaction with any amino acid in segment $\mathcal{B}_m$. In this formulation, RP is an indicator function that is set to 1 when a pair of amino acid residues make a residue pairing interaction, and 0 when an amino acid residue does not make any interaction with the opposing segment. The indices p, q, and r are used to label the amino acids either as residue pairing or non-residue pairing. In the first multiplication term, p represents the residue pairing interactions. For instance, $p = 1$ represents the first residue pairing interaction in

$\mathbf{C}_k^m$, where R_m^p, R_{m+1}^p are the interacting amino acids. Note that the residue pairing interactions can be numbered in any order. The indices q and r are used for the remaining amino acids that do not make an interaction with the opposing segment. To be more specific, q is used to label the amino acids in segment $\mathcal{B}_m$ that do not make any interaction with amino acids in $\mathcal{B}_{m+1}$. Similarly, r is used to count the amino acids in $\mathcal{B}_{m+1}$ that do not have a partner in $\mathcal{B}_m$. The range of p, q, and r depends on the number of contacts in $\mathbf{C}_k^m$. The interaction probability of a segment pair, $P(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D})$, is then computed by taking the product of the contributions from interacting amino acid pairs as well as single amino acids, which do not have any match in the opposite segment. Note that in a β-strand that interacts with two β-strands, an amino acid might not make an interaction with the previous segment but can possibly make an interaction with another amino acid in the second interacting segment.

Unfortunately, we do not have enough data to fully estimate the terms on the right hand side of Eq. (54). For this reason, we drop the dependency to $(\mathbf{O}_k, \mathbf{I}_k)$ and approximate $P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D})$ as follows

$$\begin{aligned} P(\mathbf{C}_k^m \mid \mathbf{D}) &= \prod_p P(RP = 1 \mid R_m^p, R_{m+1}^p) \times \prod_q P(RP = 0 \mid R_m^q) \quad (55) \\ &\times \prod_r P(RP = 0 \mid R_{m+1}^r) \\ P(\mathbf{C}_k \mid \mathbf{D}) &= \prod_{m=1}^{r-1} P(\mathbf{C}_k^m \mid \mathbf{D}) \quad (56) \\ P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D}) &= \frac{P(\mathbf{C}_k \mid \mathbf{D})}{\sum_{(\mathbf{C}'_k \mid \mathbf{O}_k, \mathbf{I}_k)} P(\mathbf{C}'_k \mid \mathbf{D})}. \quad (57) \end{aligned}$$

Eqs. 55, and 56 simply compute the probability of $\mathbf{C}_k$ and Eq. (57) normalizes it to obtain the conditional probability. Similar to Eq. (54), $P(RP = 1 \mid R_m^p, R_{m+1}^p)$ in Eq. (55) represents the probability of an amino acid pair to make a residue pairing interaction, and the terms $P(RP = 0 \mid R_m^q)$, $P(RP = 0 \mid R_{m+1}^r)$ represent the probability of an amino acid in one segment not to make an interaction with any

amino acid in the opposite segment. These terms can be reliably estimated from the latest available data because they do not contain any dependency to $(\mathbf{O}_k, \mathbf{I}_k)$. Similar to Eq. (53), Eq. (56) computes the contact map score of the k^{th} β-sheet.

In Eq. (57), the sum of scores is computed for all possible residue pairing patterns that are realizable for a given $(\mathbf{O}_k, \mathbf{I}_k)$. This value can be efficiently computed as

$$\sum_{(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k)} P(\mathbf{C}_k \mid \mathbf{D}) = \sum_{(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k)} \prod_{m=1}^{r-1} P(\mathbf{C}_k^m \mid \mathbf{D}) \quad (58)$$

$$= \sum_{(\mathbf{C}_k^1 \mid \mathbf{O}_k, \mathbf{I}_k)} \ldots \sum_{(\mathbf{C}_k^{r-1} \mid \mathbf{O}_k, \mathbf{I}_k)} \prod_{m=1}^{r-1} P(\mathbf{C}_k^m \mid \mathbf{D}) \quad (59)$$

$$= \prod_{m=1}^{r-1} \sum_{(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k)} P(\mathbf{C}_k^m \mid \mathbf{D}). \quad (60)$$

Eq. (58) follows from Eq. (56). In Eqs. 59 and 60, instead of sampling all possible $\mathbf{C}_k$ one by one, we sample all possible $\mathbf{C}_k^m$ for the segment pairs in $\mathbf{C}_k$ and take the product of sums to get the sum of $P(\mathbf{C}_k \mid \mathbf{D})$ values. The logic behind this approach can also be explained by the following equation, where the sums of products is converted to the product of sums.

$$\sum_i \sum_j \sum_k X_i Y_j Z_k = (\sum_i X_i)(\sum_j Y_j)(\sum_k Z_k). \quad (61)$$

The sum of the scores of all possible contact maps for a segment pair (*i.e.*, $\sum_{(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k)} P(\mathbf{C}_k^m \mid \mathbf{D})$) can be computed using dynamic programming, which is explained in Section 4.2.1.4.3.b. To illustrate how the term $P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D})$ is computed, it is useful to consider the example shown in Figure 3(a). Let the upper β-strand segment be the first segment of the sheet, which is denoted by $\mathcal{B}_1$. We need to first compute $P(\mathbf{C}_k \mid \mathbf{D})$ using Eqs. 55 and 56:

$$P(\mathbf{C}_k \mid \mathbf{D}) = P(\mathbf{C}_k^1 \mid \mathbf{D}) P(\mathbf{C}_k^2 \mid \mathbf{D}),$$

where $\mathbf{C}_k^1$ is the contact map (or the residue pairing interaction pattern) for the segment pair $(\mathcal{B}_1, \mathcal{B}_2)$, and $\mathbf{C}_k^2$ is the contact map for the segment pair $(\mathcal{B}_2, \mathcal{B}_3)$. The

terms $P(\mathbf{C}_k^1 \mid \mathbf{D})$ and $P(\mathbf{C}_k^2 \mid \mathbf{D})$ become:

$$\begin{aligned} P(\mathbf{C}_k^1 \mid \mathbf{D}) &= P(RP=1 \mid Q,V)P(RP=1 \mid D,L) \times ... \times P(RP=0 \mid G) \\ P(\mathbf{C}_k^2 \mid \mathbf{D}) &= P(RP=0 \mid V)P(RP=0 \mid L)P(RP=1 \mid I,R) \\ &\times \ ... \times P(RP=1 \mid G,C), \end{aligned}$$

where $P(RP=1 \mid Q,V)$ is the probability of the amino acid Q to make a residue pairing interaction with the amino acid V in the second segment, and $P(RP=0 \mid G)$ is the probability of the amino acid G not to make a residue pairing interaction with any amino acid in the second segment. Then, $P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D})$ can be computed using Eqs. 57- 60. In the next section, we will explain the algorithms developed for efficient computation of the optimum β-sheet conformation.

4.2.1.4 Computational Methods

The Size of the Search Space

To determine the most likely β-sheet conformation, it is necessary to search the space of conformations using efficient algorithms. There can be many alternatives for grouping β-strands into β-sheets, ordering them spatially, defining their interaction types, and matching their amino acids. The number of possible grouping combinations $\mathbf{G}$ for a protein with up to 10 β-strands is shown in Table 40. From this table, the number of possible β-sheet groups rises exponentially with the number of β-strands. In the next level, for a given grouping vector $\mathbf{G}$, we need to consider the number of

Table 40: Number of possible ways to group β-strands into β-sheets.

# β-strands	# Grouping combinations
2	1
3	1
4	6
5	11
6	89

# β-strands	# Grouping combinations
7	162
8	2140
9	8359
10	75778

ways to sample $(\mathbf{O}, \mathbf{I})$ pair, *i.e.*, the β-sheet motifs. For a β-sheet with n β-strands,

the total number of such motifs is given by $n! \times 2^{n-2}$ (the proof can be found in Ruczinski *et al* [131]). After analyzing the number of possible $(\mathbf{O}, \mathbf{I})$ values, in the third level, we should consider the number of samples we can generate for $\mathbf{C}$, *i.e.*, contact map or the amino acid residue pairing pattern of all β-sheets given the vector set $(\mathbf{G}, \mathbf{O}, \mathbf{I})$. The number of possible contact maps depends on the length of the β-strands and rises exponentially as the number of β-strands increases.

Sampling the Search Space and Computation of the Optimum Conformation

Although the number of possible conformations rises exponentially with the number of β-strands, we can reduce the computational cost by shrinking the search space to a reasonable subspace and applying efficient sampling algorithms. For the first objective, we impose β-strand segments as well as residue pairs that are predicted by the BetaPro method [39] as strong interactions. In addition, we eliminate motifs from the search space that have reasonably small motif scores. Details on space reduction methods can be found in the next two sections. For the second objective, we follow a hierarchical approach to sample the search space. We observed that if we sample the possible $\mathbf{C}$ patterns after sampling $(\mathbf{G}, \mathbf{O}, \mathbf{I})$, then we make redundant computations for some β-strand pairs. Therefore, given the amino acid sequence $\mathbf{R}$ and the secondary structure $\mathbf{SS}$, we first compute the optimum residue pairing interactions (or alignments) between all β-strand segment pairs in $\mathbf{SS}$, and store them together with their alignment scores in a table. For a protein with n_S β-strands, there are $n_S(n_S - 1)/2$ possible segment pairs. For each segment pair, we compute both parallel and anti-parallel alignments. Hence, the total number of segment alignments becomes $n_S(n_S - 1)$. The optimum alignment between two β-strand segments can be computed using the Needleman-Wunsch algorithm [103, 64], which is the global pairwise sequence alignment algorithm. Details of the Needleman-Wunsch implementation can be found in Section 4.2.1.4.3. After computing pairwise alignments of all

possible segment pairs, we sample β-sheet conformations hierarchically. We first sample $\mathbf{G}$, and for each $\mathbf{G}$, we sample $(\mathbf{O}, \mathbf{I})$. Here, we assume that the β-sheets in $\mathbf{G}$ are independent and sample possible $(\mathbf{O}_k, \mathbf{I}_k)$ values for each β-sheet separately[3]. If a particular $(\mathbf{O}_k, \mathbf{I}_k)$ combination contradicts with the significant segment pairs and their directions derived using BetaPro, then we eliminate that $(\mathbf{O}_k, \mathbf{I}_k)$ from the search space. For instance, if segments 1 and 2 have strong interaction propensity but $\mathbf{O}_k$ pairs segment 1 with segment 3, then we eliminate $\mathbf{O}_k$ from the search space. In the next step, for a given $(\mathbf{O}_k, \mathbf{I}_k)$ and $\mathbf{G}$, we simply select the best scoring residue pairing pattern $\mathbf{C}_k^*$ using the alignments we computed earlier. This is formulated as

$$\mathbf{C}_k^* = \arg\max_{\mathbf{C}_k} P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{G}, \mathbf{D}). \tag{62}$$

From Eq. (53) we have,

$$\begin{aligned} \max_{\mathbf{C}_k} P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{G}, \mathbf{D}) &= \max_{\mathbf{C}_k} \prod_m P(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{G}, \mathbf{D}) && (63) \\ &= \prod_m \max_{\mathbf{C}_k^m} P(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{G}, \mathbf{D}). && (64) \end{aligned}$$

Then, Eq. (62) can be reexpressed as

$$\begin{aligned} \mathbf{C}_k^* &= \arg\max_{\mathbf{C}_k} \prod_m P(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{G}, \mathbf{D}) && (65) \\ &= \Upsilon_m \arg\max_{\mathbf{C}_k^m} P(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{G}, \mathbf{D}) && (66) \\ &= \Upsilon_m (\mathbf{C}_k^m)^*, && (67) \end{aligned}$$

where Υ is the concatenation operator, $(\mathbf{C}_k^m)$ is the subset of $(\mathbf{C}_k)$ that defines the contact map (or the alignment) between the m^{th} β-strand pair of the k^{th} β-sheet, and $(\mathbf{C}_k^m)^*$ is the optimum contact map for that segment pair. Hence, for a given $(\mathbf{O}_k, \mathbf{I}_k, \mathbf{G})$ combination, the optimum contact map of the k^{th} β-sheet is constructed by concatenating the optimum contact maps (or the alignments) of the individual β-strand pairs (see the definition of $\mathbf{C}$ in Section 4.2.1.1).

[3] $k = 1, ..., r$, where r is the number of β-sheets in $\mathbf{G}$.

After computing the optimum contact map for a given $(\mathbf{O}_k, \mathbf{I}_k, \mathbf{G})$, we select the best scoring ordering and interaction pattern $(\mathbf{O}_k^*, \mathbf{I}_k^*)$ for the k^{th} β-sheet as

$$(\mathbf{O}_k^*, \mathbf{I}_k^*) = \arg \max_{(\mathbf{O}_k, \mathbf{I}_k)} P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{G}, \mathbf{D}) P(\mathbf{C}_k^* \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{G}, \mathbf{D}) \tag{68}$$

Let $\mathbf{C}_k^{**}$ be the optimum contact map for $(\mathbf{O}_k^*, \mathbf{I}_k^*)$. In other words, $\mathbf{C}_k^{**}$ is the optimum among $\mathbf{C}_k^*$ values derived for each $(\mathbf{O}_k, \mathbf{I}_k)$. In the next step, we can combine the optimum ordering, interaction and contact map of all β-sheets and obtain $(\mathbf{O}^*, \mathbf{I}^*, \mathbf{C}^*)$ for a given $\mathbf{G}$

$$(\mathbf{O}^*, \mathbf{I}^*, \mathbf{C}^*) = \Upsilon_{k=1}^{r} (\mathbf{O}_k^*, \mathbf{I}_k^*, \mathbf{C}_k^{**}) \tag{69}$$

Finally, the best scoring grouping $\mathbf{G}^{max}$ and the best scoring contact map $\mathbf{C}^{max}$ can be found as

$$\mathbf{G}^{max} = \arg \max_{\mathbf{G}} P(\mathbf{G} \mid \mathbf{D}) P(\mathbf{O}^*, \mathbf{I}^* \mid \mathbf{G}, \mathbf{D}) P(\mathbf{C}^* \mid \mathbf{O}^*, \mathbf{I}^*, \mathbf{G}, \mathbf{D}) \tag{70}$$

$$(\mathbf{O}^{max}, \mathbf{I}^{max}, \mathbf{C}^{max}) = \arg \max_{(\mathbf{O}^*, \mathbf{I}^*, \mathbf{C}^*)} P(\mathbf{O}^*, \mathbf{I}^* \mid \mathbf{G}^{max}, \mathbf{D}) P(\mathbf{C}^* \mid \mathbf{O}^*, \mathbf{I}^*, \mathbf{G}^{max}, \mathbf{D})$$

The algorithm for finding the optimum β-sheet conformation is summarized in Algorithm 4.

To reduce the number of computations, we applied various constraints and eliminated the low scoring conformations. In the next two sections, we explain space reduction techniques in more detail. Then, we explain how we compute the best scoring alignment between a pair of β-strands.

Constraint Based Reduction of the Search Space

To sample possible grouping combinations (*i.e.*, $\mathbf{G}$ values), we utilize a simple recursive algorithm and perform an exhaustive search. Similarly, for each β-sheet in $\mathbf{G}$, we sample every possible β-sheet motif, *i.e.*, $(\mathbf{O}, \mathbf{I})$ combinations. If the likelihood of a motif is less than the motif threshold ($P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{G}, \mathbf{D}) < t_1$), then we eliminate that motif from the search space and do not make any further computations. We chose $t_1 = 1e - 20$, a number close to zero to eliminate unlikely motifs. This approach

Algorithm 4: Computation of the Optimum β-Sheet Conformation

Input: Amino acid sequence $\mathbf{R}$, secondary structure $\mathbf{SS}$, BetaPro's residue pairing probability matrix PP, Bayesian model.

Output: Optimum β-sheet Conformation: $(\mathbf{G}^{max}, \mathbf{O}^{max}, \mathbf{I}^{max}, \mathbf{C}^{max})$

Extract β-strand segments and residue pairs with strong interaction propensities from PP;
Compute optimum pairwise alignments of β-strand segments both in parallel and anti-parallel orientation. Impose amino acid pairs with strong interaction propensities derived in step 1;
maximum overall score = 0;
for *each* $\mathbf{G}$ **do**
    grouping score = $P(\mathbf{G} \mid \mathbf{D})$;
    for *each* β*-sheet in* $\mathbf{G}$ **do**
        maximum joint score$_k$ = 0;
        for *each* $(\mathbf{O}_k, \mathbf{I}_k)$ *in the* β*-sheet* **do**
            if $(\mathbf{O}_k, \mathbf{I}_k)$ *contradicts with the significant segment pairs and their directions derived in step 1* **then**
                continue with the next $(\mathbf{O}_k, \mathbf{I}_k)$;
            motif score = $P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{G}, \mathbf{D})$;
            if *(motif score < motif threshold)* **then**
                continue with the next $(\mathbf{O}_k, \mathbf{I}_k)$;
            else
                Find $\mathbf{C}_k^*$, the optimum contact map of the β-sheet for a given $(\mathbf{O}_k, \mathbf{I}_k)$ using the table of alignments computed earlier.
                joint score = $P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{G}, \mathbf{D}) \times P(\mathbf{C}_k^* \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{G}, \mathbf{D})$;
                if *(joint score > maximum joint score*$_k$*)* **then**
                    $(\mathbf{O}_k^*, \mathbf{I}_k^*) = (\mathbf{O}_k, \mathbf{I}_k)$;
                    maximum joint score$_k$ = joint score;
                    $\mathbf{C}_k^{**} = \mathbf{C}_k^*$;
        all sheets score = $\prod_k$ maximum joint score$_k$;
        $(\mathbf{O}^*, \mathbf{I}^*, \mathbf{C}^*) = \Upsilon_{k=1}^{r}(\mathbf{O}_k^*, \mathbf{I}_k^*, \mathbf{C}_k^{**})$;
        overall score = grouping score $\times$ all sheets score;
        if *(overall score > maximum overall score)* **then**
            maximum overall score = overall score;
            $\mathbf{G}^{max} = \mathbf{G}$;
            $(\mathbf{O}^{max}, \mathbf{I}^{max}, \mathbf{C}^{max}) = (\mathbf{O}^*, \mathbf{I}^*, \mathbf{C}^*)$;

allows us to reduce the set of candidate conformations. The same approach can also be applied when sampling the $\mathbf{G}$ values, particularly when the number of β-strands is reasonably high.

Search Space Reduction using BetaPro

To further reduce the space of configurations, we found it useful to utilize the amino acid pairs predicted by BetaPro [39] with significant scores. BetaPro generates a pairing probability matrix, for all β-strand residue pairs using secondary structure, solvent accessibility and PSSM profiles, and information. In this table, each entry is a real value in the range [0, 1] and represents the propensity of an amino acid pair to make a contact. If the total number of amino acids that are labeled as β-strands is n_R, then the size of the pairing probability matrix becomes $n_R \times n_R$. We observed that when the residue pairing score is above a certain threshold, then with high confidence there is a contact between the pair. Let $S_{res-pair}$ denote the residue pairing score for a pair of amino acid residues. We consider two categories: (1) high scoring residue pairs ($S_{res-pair} > 0.16$); (2) mid scoring residue pairs ($0.02 < S_{res-pair} \leq 0.16$).[4]

We applied the following heuristics before aligning the β-strand segments. For each segment pair, we first select the corresponding sub-block from the BetaPro's pairing probability matrix and identify whether the segments form a significant pair. To align the i^{th} and j^{th} segments, we choose the sub-array in the pairing probability matrix where the rows of the sub-array correspond to the i^{th} segment and columns to the j^{th} segment. The size of this block becomes $n_r \times n_c$, where n_r and n_c are equal to the number of amino acid residues in the i^{th} and j^{th} segments, respectively. Then, we search the diagonals of the sub-block (both in parallel and anti-parallel directions) and check if there is a high or mid scoring residue pair (see Figure 13). If the number of high scoring residue pairs in a diagonal is greater than equal to two, then we flag the segment pair as high scoring and store it in a table. If the total number of high

[4]All the thresholds used in this section are found empirically.

scoring residue pairs in all diagonals is less than two, then we check if there is a mid-scoring residue pair. Similar to the high scoring case, we search the diagonals of the sub-block and identify mid-scoring residue pairs. If the number of mid scoring residue pairs in a diagonal is greater than equal to three and if the average score of such pairs is greater than or equal to 0.08, then we flag the segment pair as mid scoring and store it in a table. The average score for a set of amino acid pairs is computed simply as the sum of the residue pairing scores divided by the total number of residue pairs.

After assigning a segment pair to the high or mid scoring category, we select the high and mid scoring residue pairs for those segments. If the segment pair is in the high scoring category, we first find the diagonal that has the highest average residue pairing score and select the significant residue pairs on that diagonal. Then, we eliminate the diagonals that share the same rows and columns with the best scoring diagonal. Finally, we select the diagonals that are immediate neighbors of the best diagonal. The steps of the selection process is illustrated in Figure 14. This approach ensures that each amino acid residue makes at most one contact with the partner β-strands and allows gapped alignments. For example, the diagonals (a) and (d) in Figure 14 should generate the alignment shown in Figure 15.

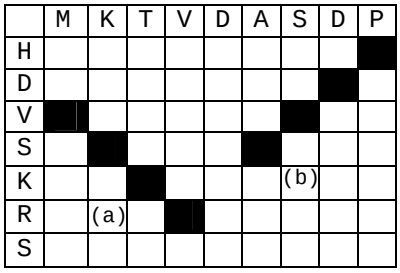


Figure 13: A sub-block of the BetaPro's residue pairing probability matrix. Each entry represents the probability of an amino acid pair to make a contact. In this figure, the segments being compared are HDVSKRS and MKTVDASDP. Diagonals of the sub-block are searched for high and mid scoring residue pairs: (a) a diagonal in parallel direction, (b) a diagonal in anti-parallel direction.

For mid scoring residue pairs, we scan the diagonals of the sub-block (both in parallel and anti-parallel directions) and store the residue pairs for which the average diagonal score is the highest (see Figure 16). Here, we do not consider a second neighboring diagonal because for mid-scoring segments the residue pairing probabilities take lower values and hence the signal to noise ratio is smaller. However, we still allow gapped alignments for the mid-scoring case. The only difference is gapped alignments are not imposed by residue pairs derived from BetaPro as in the high scoring case. The average score of a diagonal is again computed as the sum of the mid scoring residue pairs on that diagonal divided by the total number of residue pairs.

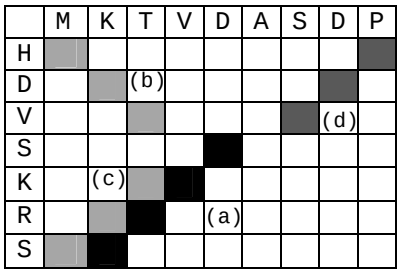


Figure 14: Identifying high-scoring residue pairs for a high scoring segment pair. (a): The diagonal with the best average residue pairing score. (b) and (c): Diagonals that are eliminated for sharing the same rows and columns with the best scoring diagonal. (d): A neighbor of the top scoring diagonal. The selected residue pairs are: H-P, D-D, V-S, S-D, K-V, R-T, S-K.

After storing segment and residue pairs with significant scores, we sort the segment pairs according to the average residue pair score. Then, we eliminate segment pairs that contribute to a cycle using a simple cycle detection algorithm from the graph theory. This step is necessary because our model does not cover β-barrels which are characterized by cyclic segment graphs. As an example to a cyclic pairing graph we

```
H D V - S K R S -
| | |   | | | |
P D S A D V T K M
```

Figure 15: The alignment expected from the high scoring residue pairs for the sub-block of the example pairing probability matrix.

can consider the following segment pairs 1-2, 2-3, 3-1, in which the segments 1 to 3 form a cyclic interaction graph. Our cycle elimination algorithm is as follows. We first check if the stored segment pairs form a cycle. This could be achieved using a simple cycle detection algorithm [109]. If there is a cycle, then we remove a segment pair with the lowest average residue pair score and check for cycles again. If there is no cycle, we terminate. If there is still a cycle, then we insert the removed segment pair back to the table and remove the second lowest segment pair. We continue until no cycle condition is satisfied. If no cycle condition is not satisfied by removing a single segment pair, then this means that there is more than one cycle. In that case, we explicitly identify the cycles including their edges and vertices and remove from each cycle the lowest scoring segment pair. Details of the heuristics applied in this section is summarized in Algorithm 5.

After identifying segments and residue pairs that are imposed in subsequent steps, we align every possible segment pair considering the residue pairs with significant scores. This is explained in the next section.

Pairwise Alignment of Segments using the Needleman-Wunsch Algorithm

We used the Needleman-Wunsch algorithm [103, 64] to compute the optimum alignment between a pair of β-strand segments. The classical implementation of the algorithm uses dynamic programming and consists of three steps: (1) initialization, (2) forward pass, (3) backtracking. In Needleman-Wunsch algorithm, the score of a

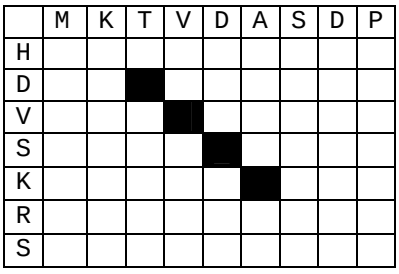


Figure 16: Identifying mid-scoring residue pairs for a mid-scoring segment pair. Only the residue pairs on the diagonal that have the highest average score are selected.

Algorithm 5: Selecting The Significant β-Strand Segments and Residue Pairs

Input: β-strand segments (segment of amino acids) in **SS** and BetaPro's residue pairing probability matrix PP. Number of segments is n_S.
Output: β-strand segments and residue pairs with strong interaction propensities.

```
for i = 1 : n_S do
    for j = 1 : n_S do
        if i = j then
            continue;
        else
            Extract the (i, j)^th sub-block of PP;
            Count the number of high scoring residue pairs (n_high) in parallel
            and anti-parallel diagonals of the sub-block;
            if (∃ a diagonal with n_high ≥ 2) then
                Flag (i, j) as a high scoring segment pair;
                Select the high scoring residue pairs;
            else
                Count the number of mid scoring residue pairs (n_mid) in
                parallel and anti-parallel diagonals of the sub-block;
                if (∃ a diagonal with n_mid ≥ 3) AND (average score > 0.08)
                then
                    Flag (i, j) as a mid scoring segment pair;
                    Select the mid scoring residue pairs;
                else
                    continue;
Drop segment pairs that are part of a cycle. Start eliminating the segment
pairs with the lowest average score;
```

path is computed by adding match or gap scores since they are essentially log-odds values. For the β-strand alignment problem, we used a similar approach. We first initialized the dynamic programming matrix at position $(0, 0)$ to 0. We then set $s(i, j)$ to $\log P(RP = 1 \mid R_i, R_j)$, which is the match/mismatch score for aligning the amino acid R_i to R_j (see Section 4.2.1.3). For gap scores, we chose $d(i)$ as $\log P(RP = 0 \mid R_i)$ and $d(j)$ as $\log P(RP = 0 \mid R_j)$, where $d(i)$ is the gap penalty score for aligning the i^{th} amino acid of the first sequence to a gap symbol, and $d(j)$ is the gap score for aligning the j^{th} amino acid of the second sequence to a gap symbol. The dynamic programming matrix is then computed by adding the match/mismatch and gap scores

as formulated in Eqs. 71 to 73.

$$M(i,0) = M(i-1,0) + d(i) \quad 1 \leq i \leq l_1 \tag{71}$$

$$M(0,j) = M(0,j-1) + d(j) \quad 1 \leq j \leq l_2 \tag{72}$$

$$M(i,j) = \max \begin{cases} M(i-1,j-1) + s(i,j) \\ M(i-1,j) + d(i) \\ M(i,j-1) + d(j) \end{cases} \tag{73}$$

After computing the dynamic programming matrix, we start from the cell indexed as (l_1, l_2), and perform backtracking to find the optimum alignment path. For this purpose, we used the same backtracking algorithm as in the classical implementation of the Needleman-Wunsch algorithm [53, 103, 64]. The alignment score is then converted to a probability value by computing its exponential.

Enforcing High and Mid Scoring Residue Pairs in the Alignment

As explained in Section 4.2.1.4, the alignment between a pair of β-strand segments is computed using the Needleman-Wunsch algorithm. After identifying high and mid scoring residue pairs, we need to make sure that the optimum alignment path passes through such pairs. This can be achieved by a simple modification of the Needleman-Wunsch algorithm. Let the β-strand segments that will be aligned have l_1 and l_2 amino acids, respectively. Also, let the m^{th} amino acid of the first segment and the n^{th} amino acid of the second segment have a significant residue pairing probability score. In the classical implementation of the Needleman-Wunsch algorithm, first, a dynamic programming matrix, which contains the alignment scores of sub-paths up to a certain residue pair is computed in the forward pass. Since our alignment should pair the m^{th} amino acid of the first segment to the n^{th} amino acid of the second segment, we need to make sure that the alignment path makes a transaction from $(m-1, n-1)$ to (m, n). This can be easily guaranteed by setting the scores of the cells $(m,0), (m,1), ..., (m,n-1)$ and $(0,n), (1,n), ..., (m-1,n)$ to 0 as shown in Figure 17

during the forward pass. When this step is repeated for all residue pairs in the high or mid scoring category, they are guaranteed to appear in the resulting alignment.

The Sum of the Alignment Scores

In Eq. (60), for each segment pair in a given β-sheet, the sum of the alignment scores of all possible residue pairing combinations has to be computed. This can be performed efficiently using a dynamic programming approach. Let M_{sum} denote a dynamic programming matrix, similar to the M matrix used in the Needleman-Wunsch algorithm. The only difference is that M_{sum} includes the sum of the scores of alignment paths instead of the maximum scores. The initialization of M_{sum} is the same as that of the M matrix. On the other hand, the forward pass equation takes the following form:

$$M_{sum}(i,j) = \log(e^{M_{sum}(i-1,j-1)+s(i,j)} + e^{M_{sum}(i-1,j)+d(i)} + e^{M_{sum}(i,j-1)+d(j)}), \qquad (74)$$

where e is the exponential. Therefore, at each position, instead of choosing the maximum score, we compute the sum of scores. Then, the sum of the scores of all possible alignments expressed as $\sum_{(\mathbf{C}_k^m \mid \mathbf{O}_k, \mathbf{I}_k)} P(\mathbf{C}_k^m \mid \mathbf{D})$ becomes equal to $exp(M_{sum}(l_1, l_2))$. This can be easily proved using Eq. (61), which is omitted here for simplicity.

Computation Times

	W	Y	L	I	T	E	S
A				0			
K				0			
V	0	0	0	↘			
D							
Q							

Figure 17: Modification of the dynamic programming matrix during the forward pass of the Needleman-Wunsch algorithm. The segments being aligned are AKVDQ and WYLITES. The amino acid residues V and I are detected as a significant residue pair. To ensure the alignment path matches V to I, the cells shown are assigned to zero. This discards all the paths that do not pair V with I.

The BetaPro method has three modular blocks. The first block generates a pairing probability matrix using the amino acid sequence, secondary structure, solvent accessibility and PSSM profiles. The second and third blocks compute the optimum β-sheet conformation by dynamic programming. Computationally, the first block is more intensive as compared to the second and third blocks due to the derivation of PSSM profiles using the PSI-BLAST algorithm. On average, the last two blocks take at most a couple of seconds to execute, whereas the first block's execution time is on the order of minutes.

Our method uses the pairing probability matrix of BetaPro to extract the amino acid pairs that have strong interaction propensities. Therefore, we first execute the first block of BetaPro and then, we sample possible conformations using efficient algorithms. Since we reduce the space of conformations significantly through the utilization of BetaPro's pairing probability matrix, our computations are significantly reduced. On average our method computes the optimum conformation of a protein with six or less β-strands in 0.31 seconds. For proteins with four β-strands it takes approximately 1 second to compute the optimum conformation. This is the same for proteins with five or six β-strands. Therefore, our method is computationally efficient and the computation time does not rise exponentially with the number of β-strands. Note that we implemented our method on a Windows XP OS, with an Intel Pentium III Xeon processor, 930 MHz CPU and 640MB RAM. BetaPro and PSI-BLAST on the other hand are implemented on a 32-bit GNU/Linux machine with Intel Pentium IV processor, 3.0 GHz CPU and 2GB RAM.

4.2.2 Beta-Sheet Prediction for Proteins with $>$ 6 Beta-Strands

The Bayesian nature of the Ruczinski model requires sufficient amount of training data to reliably estimate probability distributions. As the number of β-strands increase, the number of possible motifs rise exponentially. For proteins with more than

four β-strands, Ruczinski model reduces the feature set (or dimensions) by grouping proteins according to their structural properties. In our simulations we observed that, for proteins with more than six β-strands, the model becomes less specific and therefore its discriminative power reduces (the result not shown). For such proteins, instead of utilizing a Bayesian approach, we simply choose the same β-strand pairing predictions as BetaPro. Then, we compute gapped alignments of the paired β-strands both in parallel and anti-parallel directions. Here, for simplicity, we set the gap scores to zero and compute the score of an alignment by taking the sum of the residue pairing probability values derived using BetaPro. Finally, we select the interaction type and the residue pairing patterns with maximum alignment scores.

4.2.3 Datasets

4.2.3.1 CulledPDB

The CulledPDB set is compiled from the PDB [15] by the Dunbrack lab [7]. In this thesis and in the work by Ruczinski *et al.* [131] the set with sequence identity percentage cut-off 25% and resolution cut-off 2.5Å is used. Since the CulledPDB lists are updated periodically, the datasets grow in time. Therefore the version used by Ruczinski *et al.* is smaller in size (approximately 2000 non-homologous chains) than the one we downloaded in May 2007, which contains 2234 chains. The latest version of this dataset can be obtained from the PISCES server [14].

4.2.3.2 BetaSheet916

The BetaSheet916 set is extracted from the PDB as of May 2004 by Cheng and Baldi [39]. This dataset contains 916 chains with an HSSP threshold of 0, which corresponds to a sequence identity of 15-20%. The set is splitted randomly and evenly into 10 folds (subsets) to perform cross validation. Details of how the set is compiled can be found in Cheng and Baldi [39] and the set can be downloaded from [2].

4.2.4 BetaPro and PSI-BLAST

We downloaded and installed the BetaPro method from [2]. BetaPro uses PSI-BLAST version 2.2.8 to generate PSSM profiles. In our simulations, we used the latest versions of the PSI-BLAST (version 2.2.18) and the NR database (as of July 2008), which are obtained from the NCBI's archives [12].

4.3 Results and Discussion

4.3.1 Accuracy Measures

To assess the prediction performance, we used the sensitivity (TP/(TP+FN)) and the positive predictive value (TP/(TP+FP)) as the accuracy measures. We evaluated the predictions in the following categories: β-strand pairing, pairing direction (parallel or anti-parallel), and amino acid residue pairing (contact map). In each category, we computed the sensitivity and positive predictive value measures separately. For instance, the contact map sensitivity is computed as the total number of correctly predicted amino acid pairs divided by the total number of amino acid pairs in the dataset.

4.3.2 Experimental Settings

For the BetaPro method, we used the greedy graph algorithm to predict the β-sheet topology. Similar to the paper by Cheng and Baldi [39], we used true (native) secondary structure assignments and solvent accessibility measures, which are available in the DSSP database [11]. Hence, the results reported in this work serve as an upper bound on the performance obtained by predicted versions of secondary structure and solvent accessibility.

4.3.2.1 Model Training

The following distributions were learned from the training data: grouping distribution $P(\mathbf{G} \mid \mathbf{D})$, motif distribution $P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{D})$, and contact map distribution

$P(\mathbf{C}_k \mid \mathbf{O}_k, \mathbf{I}_k, \mathbf{D})$. The parameters used in modeling the grouping and motif distributions were estimated by Ruczinski [130] from the Culled PDB database, which is a database of non-homologous proteins (see Section 4.2.3.1). In the Culled PDB release used by Ruczinski, there were 1602 two stranded β-sheets, and 872 four stranded β-sheets (the number of three stranded β-sheets is not provided). Out of 96 possible four stranded motifs, Ruczinski observed only 48 motifs in the database. Among those, 18 motifs were observed only once and less than 20 motifs were observed ten times or more. Ruczinski used 8 bins for two stranded, 96 bins for three stranded and 1536 bins for four stranded β-sheets to estimate the probability distributions of motifs conditioned on the helical status and the connector lengths state between β-strands. Therefore, each bin represents a different configuration (or folding topology) including the motif type, helical status and connector lengths state. Ruczinski also used pseudo-counts and performed bin collapsing when the number of counts in bins were significantly low. This prevents the model to overfit to particular configurations. In the following sections, we provide details on the estimation of the parameters in our model.

Grouping Distribution

We used the same parameters as in Ruczinski [130] for $P(\mathbf{G} \mid \mathbf{D})$. We computed the term $\#[crossings(SD, n_{SH}, n_S)]$ in Eq. (9.13) of Ruczinski [130] using the Culled PDB dataset as it was not available in [130].

Motif Distribution

We used the estimated values in [130, 131] for proteins with two and three β-strands. For four stranded proteins, only the frequency information of the most common motifs is available in [130, 131]. Here, we used those frequencies as the probability values of the most common motifs and we assigned equal conditional probabilities to the remaining less common motifs. For example, the most frequent motif for a non-helical protein having short connectors (column L1 of Figure 9.9(b)

in [130] or column L1 of Figure 3 in [131]) was $\mathbf{O}$ =(1-2-3-4) and $\mathbf{I} = (AP, AP, AP)$. In our model, the probability of this motif is represented by $P(\mathbf{O}_k, \mathbf{I}_k \mid H = 0, L = (SSS))$ (see Eq. (51)) and this probability was estimated by Ruczinski as 0.85. To the remaining 95 possible motifs, we assigned equal conditional probability values *i.e.*, $P(\mathbf{O}_k, \mathbf{I}_k \mid H = 0, L = (SSS)) = 0.15/95$. For proteins with a higher number of β-strands, we estimated the parameters $P(P_p, J \mid n, H, L, F)$ and $k_{n,L}(P_p, P_p^s, J, J^s, F)$ (see Appendix A.2) using the CulledPDB dataset as of May 2007 (see Section 4.2.3.1) as they were not available in [130]. For $P(F \mid H, L)$ and $P(P_p^s, J^s \mid n, H, L, F, P_p, J)$, we used the estimated values in Ruczinski [130].

Contact Map Distribution

Contact map distribution depends on the parameters $P(RP = 1 \mid R_m^p, R_{m+1}^p)$, $P(RP = 0 \mid R_m^q)$, $P(RP = 0 \mid R_{m+1}^r)$ in Eq. (55). In this section, we estimated those probability distributions from the BetaSheet916 dataset (see Section 4.2.3.2) for which, the secondary structure assignments are taken from the DSSP database [11]. In the cross validation experiment, we only used the folds that form the training set. To estimate those parameters, we used the maximum-likelihood estimation procedure where we count the observed number of occurrences, and apply a proper normalization factor to compute probability values.

4.3.3 10-Fold Cross Validation on BetaSheet916

In the first set of simulations, we performed a 10 fold cross validation on the BetaSheet916 set, which contains 916 proteins extracted from the Protein Data Bank (PDB) (see Section 4.2.3.2 for details). In a cross validation experiment, at each step, a fold is selected as a test data and remaining folds form the training set. Then predictions are computed for proteins in the test set with the models trained on the training set. This process is repeated until all proteins in the original set are tested. Once the predictions are complete, then prediction accuracy is computed.

4.3.3.1 Performance for Proteins with ≤ 4 β-Strands

In this simulation, we performed a 10 fold cross validation experiment on BetaSheet916 and evaluated BetaZa (our method) and BetaPro for proteins with less than or equal to four β-strands. In each fold of the BetaSheet916, we only considered proteins with less than or equal to four β-strands (a total of 67 proteins). Furthermore, since the current version of our model allows up to two β-strand partners for proteins with six or less β-strands, we eliminated proteins that had β-strand segments interacting with more than 2 segments. Among the 67 proteins, there was only one protein with more than 2 segmental partner. Therefore, the total number of proteins tested from all folds becomes 66, which contain a total of 163 β-strand pairs and 1846 β-residue pairs.

Comparing the performances of BetaPro and BetaZa, we obtained the results summarized in Table 41, for sensitivity and positive predictive value (PPV) measures. From these results, we can conclude that BetaZa significantly outperforms BetaPro for proteins with less than or equal to four β-strands.

Table 41: Sensitivity and Positive Predictive Value measures, evaluated on the BetaSheet916 set. Proteins with four or less β-strands are used as the test data. Each β-strand has less than three segmental partners.

Measure	Sensitivity (%)			PPV (%)		
Prediction Category	Strand Pairing	Pairing Direction	Contact Map	Strand Pairing	Pairing Direction	Contact Map
BetaPro	81.595	79.755	72.264	85.807	83.871	73.702
BetaZa	90.798	88.344	82.232	90.244	87.805	81.965

4.3.3.2 Performance for Proteins with ≤ 6 β-Strands

In the next step, we extended our test set to include proteins with six or less β-strands and repeated the 10 fold cross validation experiment performed in Section 4.3.3.1. There were a total of 187 such proteins in BetaSheet916. Among those, 16 had β-strands with more than 2 segmental partners. Eliminating those, our test data

contained 171 proteins from all folds with 586 β-strand pairs and 5838 β-residue pairs.

The sensitivity and positive predictive value measures are shown in Table 42. For proteins with six or less β-strands, BetaZa is significantly more accurate than BetaPro. This is also validated by evaluating the accuracy for proteins with a fixed number of β-strands. Table 43 shows the performance for proteins with five β-strands, whereas Table 44 shows the performance for proteins with six β-strands. Although the positive predictive value measure of BetaPro is slightly better than BetaZa in segment pairing and interaction type categories, BetaZa performs better in sensitivity measure and especially in the contact map category. BetaPro's higher positive predictive value measure is caused by its tendency to generate fewer number of predictions instead of generating higher true positives.

Table 42: Sensitivity and Positive Predictive Value measures, evaluated on the BetaSheet916 set. Proteins with six or less β-strands are used as the test data. Each β-strand has less than three segmental partners.

Measure	Sensitivity (%)			PPV (%)		
Prediction Category	Strand Pairing	Pairing Direction	Contact Map	Strand Pairing	Pairing Direction	Contact Map
BetaPro	79.010	77.133	71.634	83.877	81.884	73.575
BetaZa	83.271	80.370	77.665	84.282	81.347	79.541

Table 43: Sensitivity and Positive Predictive Value measures, evaluated on the BetaSheet916 set. Proteins with five β-strands are used as the test data. Each β-strand has less than three segmental partners.

Measure	Sensitivity (%)			PPV (%)		
Prediction Category	Strand Pairing	Pairing Direction	Contact Map	Strand Pairing	Pairing Direction	Contact Map
BetaPro	80.349	78.603	74.803	88.462	86.534	78.947
BetaZa	83.843	80.349	77.865	86.099	82.511	79.181

Table 44: Sensitivity and Positive Predictive Value measures, evaluated on the BetaSheet916 set. Proteins with six β-strands are used as the test data. Each β-strand has less than three segmental partners.

Measure	Sensitivity (%)			PPV (%)		
Prediction Category	Strand Pairing	Pairing Direction	Contact Map	Strand Pairing	Pairing Direction	Contact Map
BetaPro	75.258	73.196	66.706	77.249	75.132	66.628
BetaZa	76.289	73.711	72.333	76.684	74.093	77.125

4.3.3.3 Overall Performance

In this simulation, we evaluated the accuracy on the full set of proteins by performing a 10 fold cross validation experiment on BeteSheet916. This set contains a total of 8172 β-strand pairs and 31638 β-residue pairs. For proteins with six or less β-strands, we computed the predictions as described in Section 4.2.1, and for proteins that contain more than six β-strands as in Section 4.2.2. Among proteins with six or less β-strands, we eliminated those with more than two segmental interactions (removing only 16 proteins). For the remaining proteins, we allowed a β-strand to interact with more than two segments because we used BetaPro to compute β-strand pairing predictions. Hence, the overall accuracy of BetaZa is not significantly different from that of BetaPro in the first two categories. However, due to gapped alignments of β-strands, the β-residue pairing accuracy of BetaZa is better than BetaPro by 3% both in sensitivity and positive predictive value measures.

Table 45: Sensitivity and Positive Predictive Value measures, evaluated on the BetaSheet916 set. Only 16 proteins that had: (1) ≤ 6 β-strands and (2) at least one β-strand with more than two segmental interactions are excluded from the test data.

Measure	Sensitivity (%)			PPV (%)		
Prediction Category	Strand Pairing	Pairing Direction	Contact Map	Strand Pairing	Pairing Direction	Contact Map
BetaPro	68.903	66.072	63.411	61.921	59.376	54.373
BetaZa	69.075	66.244	66.477	61.911	59.373	57.211

4.3.3.4 Performance of BetaZa for Individual Configurations

The analysis performed by Ruczinski *et al.* [131] shows that a handful of β-sheet configurations are much more frequent than the others. This means that higher probability values will be assigned to such configurations. In that case, it becomes important to verify that our method is capable of generating accurate predictions for less frequent configurations. To understand this, we analyzed the performance on individual proteins. For this purpose, we considered proteins with less than or equal to four β-strands as in Section 4.3.3.1. There are 66 proteins and 39 distinct configurations in this test data. Here, a configuration is represented by the following features: β-sheet motif (spatial ordering and interaction types), helical status of the protein, and the length states of the segments that connect β-strands. We defined a configuration as less frequent when the motif probability assigned by the model is less than 0.05. The motif frequencies can be found in Ruczinski *et al.* [131].

Table 46 shows the sensitivity and positive predictive value of individual β-sheet configurations. In each row, the features that characterize the configuration as well as the motif probabilities conditioned on the helical status and connecting lengths states are listed. In this table, the symbol "|" is used to separate β-sheets in the spatial ordering representation. For instance, 1-2|3-4 means that the first and the second β-strands form the first β-sheet; the third and the fourth β-strands form the second β-sheet; and there is no interaction between the second and the third segments. Alternatively, 1-2-3-4 shows that all four β-strands form a single β-sheet. The helical status and the connecting length states are defined in Section 4.2.1.3. Here NH stands for non-helical and H for helical protein. Similarly, S denotes a short connector and L represents a long connector. A connector is a set of helix and/or loop segments that are in between β-strand pairs adjacent in sequence representation. From this table, we can observe that although the prediction accuracy of less frequent configurations is in general lower than the frequent ones, our method was able to generate highly

accurate predictions for five configurations that have significantly low probability scores (marked in boldface). This clearly demonstrates that our method is able to predict less frequent motifs with high accuracy and the increase in the performance is not simply because of an affinity towards for more frequent motifs or an imbalance of the training data.

Table 46: Performance of BetaZa for individual configurations. Proteins with ≤ 4 β-strands are evaluated. The protein that contains a β-strand with more than two segmental interactions is excluded.

Spatial Ordering	Interaction Types	Helical Status	Connecting Lengths	Motif Probability	Strand Pairing Sensitivity (%)	Strand Pairing PPV (%)	Interaction Type Sensitivity (%)	Interaction Type PPV (%)	Contact Map (%) Sensitivity (%)	Contact Map(%) PPV (%)
1-2	A	H	S	0.9900	100.0	100.0	100.0	100.0	100.0	100.0
1-2|3-4	A|A	NH	SSS	0.9801	100.0	100.0	100.0	100.0	100.0	100.0
1-2|3-4	A|A	H	SSS	0.9801	100.0	100.0	100.0	100.0	94.118	100.0
1-2-3	A-A	H	SS	0.8970	100.0	100.0	100.0	100.0	97.753	95.604
1-2-3	A-A	NH	SS	0.8970	100.0	100.0	100.0	100.0	96.0	96.0
1-2	A	NH	L	0.8700	100.0	100.0	100.0	100.0	100.0	100.0
1-4|2-3	A|A	NH	LSL	0.8613	75.0	60.0	75.0	60.0	53.846	43.750
1-2-3-4	A-A-A	H	SSS	0.8500	100.0	100.0	100.0	100.0	87.500	87.500
1-4|2-3	A|A	H	LSL	0.7227	50.0	33.333	50.0	33.333	42.857	33.333
1-2|3-4	A|A	H	LSS	0.7227	100.0	100.0	100.0	100.0	100.0	100.0
1-3-2	A-A	H	LS	0.5472	100.0	100.0	100.0	100.0	96.429	93.103
1-2-4-3	A-A-A	H	SLS	0.5100	83.333	90.909	83.333	90.909	76.923	75.757
2-3-1-4	A-A-A	H	LSL	0.3800	100.0	100.0	100.0	100.0	76.667	76.033
2-1-3-4	P-P-P	H	LLL	0.3600	100.0	100.0	100.0	100.0	100.0	100.0
2-1-3-4	P-A-A	H	LLS	0.2800	66.667	100.0	66.667	100.0	66.667	100.0
1-2	P	H	L	0.2700	100.0	50.0	100.0	50.0	100.0	85.714
1-2|3-4	P|A	H	LSS	0.2673	100.0	100.0	100.0	100.0	100.0	100.0
1-2-3	A-P	H	SL	0.2622	100.0	100.0	100.0	100.0	92.307	92.307
2-1-3	A-A	H	SL	0.2587	50.0	50.0	50.0	50.0	46.154	54.545
1-4-2-3	A-A-A	H	LSL	0.2400	66.667	66.667	66.667	66.667	60.0	64.286

1-4-3-2	A-A-A	H	LSS	0.1800	100.0	100.0	100.0	100.0	72.500	70.732
2-3-1-4	A-A-A	NH	LSL	0.1800	100.0	100.0	100.0	100.0	84.0	75.0
2-1-3	A-A	H	LL	0.1525	100.0	100.0	100.0	100.0	83.333	83.333
1-2-4-3	A-A-A	H	LLS	0.1200	100.0	100.0	66.667	66.667	77.143	75.0
1-2-4-3	P-A-A	H	LLS	0.1000	100.0	100.0	100.0	100.0	82.692	81.132
2-3-1-4	A-A-A	H	LLL	0.0800	100.0	100.0	100.0	100.0	61.905	65.0
2-1-4-3	A-P-A	H	SLS	0.0800	100.0	100.0	66.667	66.667	78.378	76.316
1-2-3	**P-P**	**H**	**LL**	**0.0491**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
2-1-3	A-P	NH	SS	0.0279	50.0	50.0	50.0	50.0	87.500	87.500
1-2-3-4	**A-A-A**	**H**	**LSS**	**0.0090**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
2-1-4-3	A-A-A	NH	SLL	0.0049	66.667	66.667	66.667	66.667	70.588	80.0
1-4-3-2	A-A-P	H	LLL	0.0042	66.667	100.0	66.667	100.0	41.667	62.500
1-4-2-3	A-A-P	H	LLL	0.0042	66.667	66.667	66.667	66.667	66.667	72.727
1-2-3-4	**A-A-A**	**H**	**LLL**	**0.0042**	**100.0**	**100.0**	**100.0**	**100.0**	**92.307**	**88.889**
3-2-1-4	P-A-A	H	LLL	0.0042	66.667	66.667	66.667	66.667	76.923	71.429
1-3-4-2	A-A-P	H	LLS	0.0036	66.667	66.667	66.667	66.667	64.286	64.286
1-2-4-3	**A-A-P**	**H**	**SSL**	**0.0025**	**100.0**	**100.0**	**100.0**	**100.0**	**86.667**	**92.857**
1-4-3-2	A-A-A	NH	SSS	0.0015	66.667	66.667	66.667	66.667	91.667	91.667
1-3-4-2	**P-A-A**	**NH**	**SSS**	**0.0015**	**100.0**	**100.0**	**100.0**	**100.0**	**80.0**	**80.0**

4.4 Summary

In this chapter, we have shown that elaborate mathematical models combined with efficient algorithms bring significant improvements to β-sheet prediction. We addressed the problem of β-sheet prediction defined as the prediction of β-strand pairings, interaction types (parallel or anti-parallel), and β-residue interactions (or contact maps). We analyzed proteins according to the number of β-strands they contain. We introduced a Bayesian approach for proteins with six or less β-strands, in which we modeled the conformational features in a probabilistic framework by combining the amino acid pairing potentials with *a priori* knowledge of β-strand organizations. To select the optimum β-sheet architecture, we analyzed the space of possible conformations by efficient heuristics, in which we significantly reduce the search space by enforcing the amino acid pairs that have strong interaction potentials. Furthermore, we employed an algorithm that finds the optimum pairwise alignment between β-strands using dynamic programming. For proteins with more than six β-strands, we first computed β-strand pairings using the BetaPro method. Then, we computed gapped alignments of the paired β-strands in parallel and anti-parallel directions and chose the interaction types and β-residue pairings with maximum alignment scores. We performed a 10-fold cross validation experiment on the BetaSheet916 set and obtained significant improvements in the prediction accuracy in all categories for proteins with six or less β-strands. For proteins with higher number of β-strands we obtained significant improvements in the contact map prediction category with other categories yielding equal performance.

CHAPTER V

CONCLUSION

In this thesis, we developed Bayesian models and machine learning algorithms for protein secondary structure and β-sheet prediction.

In protein secondary structure prediction, we concentrated on proteins in the single-sequence category which do not share any significant similarity with any other protein. Such "orphan" proteins are difficult targets for functional characterization and necessitate the utilization of additional knowledge sources. For an orphan protein, any method of secondary structure prediction performs as a single-sequence method. Developing better methods of secondary structure prediction from single-sequence has a definite merit as it helps improving the functional annotation of orphan proteins. With this motivation, we showed that sophisticated dependency models and training methods bring further improvements to protein secondary structure prediction. As new sequences are added to the database, it will be possible to augment the dependency structure and obtain even higher accuracy.

Typically protein secondary structure prediction methods suffer from low accuracy in predicting β-strands, in which non-local correlations have a significant role. In this thesis, we developed an N-best strategy to incorporate long-range dependencies into our secondary structure prediction algorithm. Unfortunately, the incorporation of non-local interactions into the hidden semi-Markov model did not bring significant improvements in the single-sequence setting. Nevertheless, the N-best strategy is still promising for proteins with evolutionary homologues, which share a larger portion of the database. As a future work, it is possible to extend the N-best decoding approach for the case when evolutionary information in the form of multiple

alignment profiles (*e.g.* PSSM profiles) is available. The proposed N-best algorithms and techniques can also be applied to other problems that employ HMMs such as gene prediction, topology prediction for outer-membrane proteins, sequence-sequence and sequence-structure alignments, speech recognition, video scene annotation, and machine translation.

In β-sheet prediction, we developed Bayesian models and algorithms to compute the optimum β-sheet conformation given the amino acid sequence, secondary structure, solvent accessibility and PSSM profiles. The predictions can be improved even further. First of all, sophisticated methods for the estimation of the residue pairing propensities will definitely improve the accuracy and quality of the predictions. For this purpose, one can incorporate additional informative features such as HMM profiles, contact potentials, residue types, segment window information, and protein-level information [40]. In a second avenue, one can develop more elaborate models for an enhanced scoring of β-strand organizations. We introduced a Bayesian model for proteins with six or less β-strands and allowed each β-strand to interact with at most two other segments. Extension of the model to characterize higher order segmental interactions can easily be achieved by estimating their probabilities and sampling them in the search space. For proteins with more than six β-strands, it is possible to incorporate a richer set of folding rules as in [79]. Finally, as new proteins are added to the structure database it will be possible to extend the motif distribution to model longer proteins with many β-strands and extend the coverage of the Bayesian model. Advances in protein secondary structure and β-sheet prediction will contribute substantially to the accurate prediction of the function and the 3-D structure.

APPENDIX A

BAYESIAN MODELS FOR BETA-SHEET GROUPINGS AND ORDERINGS

A.1 The Grouping Term $P(\mathbf{G} \mid \mathbf{D})$

A.1.1 $P(SD \mid n_{SH}, n_S)$

To model the first component of $P(\mathbf{G} \mid \mathbf{D})$, the number of crossings is used as a surrogate. The number of crossings is defined as the number of times that one leaves a β-sheet and enters another traversing the backbone of the protein from the N-terminus to the C-terminus (sequential order). In this model, it has been assumed that given the number of strands and sheets, all decompositions that yield the same number of crossings are equally likely. For example, the decompositions 1-2-3-1-2-3 and 1-2-3-2-1-3 of six stranded proteins with three β-sheets[1] both have 5 crossings and are considered to be equally likely.

In that case, the model takes the following form:

$$P(SD \mid n_{SH}, n_S) = \frac{P(\#crossings(SD))}{\#SD^*} \tag{75}$$

if $n_{SH} \geq 2$ and 1 otherwise. In the above equation, $\#crossings(SD)$ is the number of crossings in decomposition SD and $\#SD^*$ is the total number of sheet decompositions with the same number of crossings.

If there are n_{SH} β-sheets, there can be at least $n_{SH} - 1$ and at most $n_S - 1$ crossings. It has been found that the physical nature of structure formation favors proteins with small number of crossings. Hence, it is reasonable to consider two scenarios: (1) Having the minimum number of crossings; (2) Having the number of

[1] Numbers represent β-sheets.

crossings in excess of the minimum. Next, we explain the models derived for each case in more detail.

(1) $\#crossings(SD) = n_{SH} - 1$: In this scenario, the outcome is binary (having the minimum number of crossings versus not having the minimum). Therefore it is possible to use logistic regression to predict this outcome. Based on the available data, it is useful to distinguish proteins with two β-sheets from proteins with more than two β-sheets.

$$\log \frac{p}{1-p} = -3.372 + 0.653 \times n_s - 1.285 \times I_{(n_{SH}>2)} \tag{76}$$

with p being equal to $P(\#crossings(SD) = n_{SH} - 1)$, and I being the indicator function of the argument taking the value of one if $n_{SH} > 2$ and zero otherwise.

(2) $\#crossings(SD) > n_{SH} - 1$: Let E_{max} be the maximum number of crossings by which we can exceed $n_{SH} - 1$. Then we can define the variable Y as

$$Y = \#crossings - n_{SH}, \tag{77}$$

where $Y \in \{0, ..., E_{max} - 1\}$. Ruczinski [130] approximated the distribution of Y using a Poisson model:

$$P(Y = k) = \frac{\exp(-\lambda)\frac{\lambda^k}{k!}}{\sum_{j=0}^{E_{max}-1} \exp(-\lambda)\frac{\lambda^j}{j!}} \tag{78}$$

where $P(Y = k)$ is Poisson for $k < E_{max}$ and 0 otherwise. In this model, the parameter λ can be estimated as

$$\log(\lambda) = -1.185 + 0.195 \times n_S - 0.463 \times I_{(n_{SH}>2)}, \tag{79}$$

which is equivalent to

$$\lambda = 0.306 \times 1.215^{n_S} \times 0.629^{I_{(n_{SH}>2)}}. \tag{80}$$

To summarize the model for the number of crossings, let:

X	be the number of crossings in excess of $n_{SH} - 1$
Y	be the number of crossings in excess of n_{SH}, *i.e.*, $Y = X - 1$
Z	be an indicator if the number of crossings exceeds $n_{SH} - 1$
$logistic$	be the term on the right hand side of Eq. (76)
$poisson$	be the term on the right hand side of Eq. (78)

Then, we have

$$P(X = 0) = P(Z = 0) = \frac{\exp(logistic)}{1 + \exp(logistic)}, \tag{81}$$

and for $j \in \{1, ..., E_{max}\}$ we get

$$\begin{aligned} P(X = j) &= P(X = j, Z = 1) \\ &= P(X = j \mid Z = 1)P(Z = 1) \\ &= P(Y = j - 1 \mid Z = 1)P(Z = 1) \\ &= poisson \times \frac{1}{1 + \exp(logistic)}. \end{aligned}$$

A.1.2 $P(n_{SH} \mid n_S)$

Since every β-sheet has to have at least two β-strands, the maximum number of β-sheets is

$$n_{S_{max}} = [\frac{n_S}{2}]. \tag{82}$$

Let X be the number of β-sheets in excess of the one β-sheet required, and define $n := n_{S_{max}} - 1$, where $X \in \{0, ..., n\}$. Analyzing the data, X can be modeled as a binomial distribution assuming

$$X \sim B(n, p(n_S)). \tag{83}$$

Ruczinski [130] found that the probability in the binomial distribution does not depend on the number of β-strands, and estimated

$$p(n_S) \equiv p = 0.35. \tag{84}$$

A.2 Modeling of $P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{D})$ for proteins with more than four β-strands

The feature set of the k^{th} β-sheet, Λ_k, consists of the following components:

P_p : Number of parallel β-strands in a motif

P_p^s : Number of parallel β-strands with a short connector in between

J : Number of β-strand pairs adjacent in sequence that are not neighbors in the β-sheet (*i.e.*,"jumps")

J^s : Number of jumps with a short connector in between

F : The position of the first β-strand in the β-sheet.

With this representation, we can model the term $P(\Lambda_k \mid H, L)$ as

$$\begin{aligned} P(\Lambda_k \mid H, L) &= P(P_p, P_p^s, J, J^s, F \mid H, L) \\ &= P(F \mid H, L) P(P_p, J \mid F, H, L) P(P_p^s, J^s \mid F, P_p, J, H, L). \end{aligned} \tag{85}$$

Next, we will concentrate on each of these terms and present modeling assumptions.

A.2.1 $P(F \mid H, L)$

Analyzing the data, it is possible to make the following assumption

$$P(F \mid H, L) = P(F \mid n, H), \tag{86}$$

where n is the number of β-strands in the β-sheet, and H is the helical status of the protein. Note that a protein is labeled as helical if at least 20% of its amino acids are part of an α-helix, and non-helical otherwise. $P(F \mid n, H)$ is estimated from the available data in Ruczinski [130].

A.2.2 $P(P_p, J \mid F, H, L)$

Similar to the previous section, we assume that

$$P(P_p, J \mid F, H, L) = P(P_p, J \mid F, n, H, L). \tag{87}$$

For a fixed n, there are n possibilities for P_p, n possibilities for J, 2 possibilities for H, 2^{n-1} possibilities for L, and $[\frac{n+1}{2}]$ possibilities for F. Therefore, there are approximately $2^{n-1} \times n^3$ bins in the term $P(P_p, J \mid F, n, H, L)$. Ruczinski [130] performed a χ^2-test and found that P_p and J cannot be assumed to be conditionally independent. To further simplify the probability term the following criteria is applied: (1) For F, only discriminate between starting the β-sheet at the first spatial position versus starting at any other permissable position. In that case, F takes only two values: $F = 1$ when starting at the first position and $F = 2$ otherwise; (2) Certain motifs occur when all connectors between β-strands are more than ten residues long. Set $L = 1$ if all connectors between β-strands are long, and $L = 0$ otherwise. Based on these assumptions, we estimated $P(P_p, J \mid F, n, H, L)$ using the available data derived from the PDB.

A.2.3 $P(P_p^s, J^s \mid F, P_p, J, H, L)$

Analyzing the data, it reasonable to assume the following conditional independence

$$P(P_p^s, J^s \mid F, P_p, J, H, L) = P(P_p^s \mid F, P_p, J, H, L)P(J^s \mid F, P_p, J, H, L). \tag{88}$$

In the above equation, the terms in the right hand side can be further simplified by removing parameters that show weak dependency. This is formulated as

$$\begin{aligned} P(P_p^s \mid F, P_p, J, H, L) &= P(P_p^s \mid P_p, H, L), \\ P(J^s \mid F, P_p, J, H, L) &= P(J^s \mid J, H, L). \end{aligned} \tag{89}$$

Let n_p be the number of parallel pairs in the β-sheet, and n_{sc} the number of short connectors. Since there are $n-1$ pairs of β-strands in the β-sheet, the lowest possible number of parallel pairs of β-strands that are connected by a short connector is given by

$$l = \max(n_{sc} + n_p - (n-1), 0). \tag{90}$$

The maximum number of parallel pairs of β-strands in the β-sheet that are connected by a short connector is

$$u = \min(n_p, n_{sc}). \tag{91}$$

Since $n_{sc} \in \{l, ..., u\}$, we are interested in modeling the number of parallel pairs, X, that are connected by a short connector in excess of l. Hence, $X \in \{0, ..., u-l\}$. The data support the following model:

$$\begin{aligned} P(P_p^s = k + l \mid n, H, L, P_p) &= P(P_p^s = k + l \mid n, H, n_{sc}, n_p), \\ &= P(X = k) \end{aligned} \tag{92}$$

where

$$X \sim B(u - l, p_{par}(n, H)). \tag{93}$$

Analyzing the data, Ruczinski [130] observed that the probability of the binomial term does not change significantly for different β-sheet sizes. $p_{par}(n, S_H)$ is estimated as follows:

$$p_{par}(n, H) = p_{par}(H) = \begin{cases} 0.51 & \text{if } H = 0 \\ 0.24 & \text{if } H = 1. \end{cases} \tag{94}$$

For the number of jumps with a short connector, a similar approach can be used to derive the model. Let j be the number of jumps in the β-sheet and n_{sc} be the number of short connectors. Let Y be the number of short jumps in excess of the lowest possible number of jumps with a short connector. We have

$$\begin{aligned} P(J^s = k + l \mid n, H, L, J) &= P(J^s = k + l \mid n, H, n_{sc}, j) \\ &= P(Y = k), \end{aligned} \tag{95}$$

where

$$Y \sim B(u - l, p_{jump}(n, H)), \tag{96}$$

and

$$l = \max(n_{sc} + j - (n - 1), 0) \tag{97}$$

$$u = \min(j, n_{sc}). \tag{98}$$

In this case, the binomial term depends on the β-sheet size, and is derived in Ruczinski [130] as

$$p_{jump}(n, H) = p_{jump}(H) = \begin{cases} 0.25 & \text{if } H = 0 \text{ and } n = 5 \text{ or } 6 \\ 0.54 & \text{if } H = 1 \text{ and } n = 5 \text{ or } 6 \\ 0.18 & \text{if } H = 0 \text{ and } n \geq 7 \\ 0.28 & \text{if } H = 1 \text{ and } n \geq 7. \end{cases} \tag{99}$$

This completes the modeling of the motif likelihood term $P(\mathbf{O}_k, \mathbf{I}_k \mid \mathbf{D})$.

REFERENCES

[1] "3rd generation prediction of secondary structure." http://www.embl-heidelberg.de/∼rost/Papers/1999_humana/paper.html.

[2] "BetaSheet916 set." http://www.ics.uci.edu/∼baldig/betasheet_data.html.

[3] "CASP6 targets." http://predictioncenter.org/casp6/.

[4] "CB513 set." http://www.compbio.dundee.ac.uk/∼www-jpred/data/.

[5] "Critical assessment of fully automated structure prediction." http://www.cs.bgu.ac.il/∼dfischer/CAFASP3/.

[6] "Critical assessment of techniques for protein structure prediction." http://predictioncenter.llnl.gov/casp6/.

[7] "Dunbrack Lab." http://dunbrack.fcc.edu.

[8] "EVA results." http://cubic.bioc.columbia.edu/eva/cafasp/sechom/method.

[9] "EVA: secondary structure (intro)." http://cubic.bioc.columbia.edu/eva/doc/intro_sec.html.

[10] "EVA set." http://cubic.bioc.columbia.edu/eva/doc/ftp.html.

[11] "FTP access to the DSSP files at the CMBI." ftp://ftp.cmbi.kun.nl/pub/molbio/data/dssp.

[12] "NCBI BLAST Downloads." http://www.ncbi.nlm.nih.gov/BLAST/download.shtml.

[13] "PDB_SELECT dataset." http://bioinfo.tg.fh-giessen.de/pdbselect/.

[14] "Pre-compiled CulledPDB lists from PISCES." http://dunbrack.fcc.edu/Guoli/pisces_download.php#culledpdb.

[15] "The protein data bank." http://www.rcsb.org/pdb.

[16] "PSIPRED server." http://bioinf.cs.ucl.ac.uk/psipred/.

[17] "PSIPRED_v2.0 training set." http://bioinf.cs.ucl.ac.uk/downloads/psipred/old/data/.

[18] "Tailbiting decoder and method, European Software Patents." http://swpat.ffii.org/pikta/txt/ep/1258/086/#data.

[19] An, Y. and Friesner, A., "A novel fold recognition method using composite predicted secondary structure," *Proteins: Structure Function and Genomics*, vol. 48, pp. 352–366, 2002.

[20] Asai, K., Hayamizu, S., and Handa, K. I., "Prediction of protein secondary structure by the hidden Markov model," *Comp. Applic. Boiosci.*, vol. 9, no. 2, pp. 141–146, 1999.

[21] Asogawa, M., "Beta-sheet prediction using inter-strand residue pairs and refinement with hopfield neural network," in *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, vol. 5, pp. 48–51, 1997.

[22] Aurora, R. and Rose, G. D., "Helix capping," *Prot. Sci.*, vol. 7, pp. 21–38, 1998.

[23] Aydin, Z., Altunbasak, Y., and Borodovsky, M., "Protein secondary structure prediction for a single sequence using hidden semi-Markov models," *BMC Bioinformatics*, vol. 7, no. 178, 2006.

[24] Aydin, Z., Altunbasak, Y., and Erdogan, H., "Bayesian protein secondary structure prediction with near-optimal segmentations," *IEEE Trans. Signal Proc.*, vol. 55 (Issue 7), pp. 3512–3525, 2007.

[25] Aydin, Z., Erdogan, H., and Altunbasak, Y., "Protein fold recognition using residue-based alignments of sequence and secondary structure," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Proc. (ICASSP'07)*, vol. 1, pp. I349–352, 2007.

[26] Bagos, P. G., Liakopoulos, T. D., Spyropoulos, I. C., and Hamodrakas, S. J., "A hidden markov model method capable of predicting and discriminating β-barrel outer membrane proteins," *BMC Bioinformatics*, vol. 5, no. 29, 2004.

[27] Bahl, L. R. and Jelinek, F., "Apparatus and method for determining a likely word sequence from labels generated by an acoustic processor," *US Patent, 4,748,670*, May 1988.

[28] Baldi, P., Brunak, S., Frasconi, P., Pollastri, G., and Soda, G., "Exploiting the past and future in protein secondary structure prediction," *Bioinformatics*, vol. 15, pp. 937–946, 1999.

[29] Baldi, P., Pollastri, G., Andersen, C. A. F., and Brunak, S., "Matching protein β-sheet partners by feedforward and recurrent neural networs," in *Proc. Int. Conf. Intell. Syst. Mol. Biol. (ISMB'00)*, vol. 8, pp. 25–36, 2000.

[30] Bau, D., Martin, A., Mooney, C., Vullo, A., Walsh, I., and Pollastri, G., "Distill: a suite of web servers for the prediction of one-, two- and three-dimensional structural features of proteins," *BMC Bioinformatics*, vol. 7, no. 402, 2006.

[31] Bienkowska, J. R., Yu, L., Zarakhovich, S., Rogers, R. G. J., and Smith, T. F., "Protein fold recognition by total alignment probability," *Proteins*, vol. 40, no. 3, pp. 451–462, 2000.

[32] Bindewald, E., Cestaro, A., Hesser, J., Heiler, M., and Tosatto, S. C. E., "MANIFOLD: Protein fold recognition based on secondary structure, sequence similarity and enzyme classification," *Protein Engineering*, vol. 16, no. 11, pp. 785–789, 2003.

[33] Borodovsky, M. and Lukashin, A. V., "GeneMark.hmm: new solutions for gene finding," *Nucleic Acids Res.*, vol. 26, pp. 1107–1115, 1998.

[34] Bradley, P., Cowen, L., Menke, M., King, J., and Berger, B., "BETAWRAP: Successful prediction of parallel β-helices from primary sequence reveals an association with many pathogens," *PNAS*, vol. 98, pp. 14819–14824, 2001.

[35] Burge, C. and Karlin, S., "Prediction of complete gene structures in human genomic DNA," *J. Mol. Biol*, vol. 268, pp. 78–94, 1997.

[36] Bystroff, C., Thorsson, V., and Baker, D., "HMMSTR: a hidden markov model for local sequence structure correlations in proteins," *J. Mol. Biol.*, vol. 301, pp. 173–190, 2000.

[37] Cawley, S. L. and Pachter, L., "HMM sampling and applications to gene finding and alternative splicing," *Bioinformatics*, vol. 19 (Suppl. 2), pp. ii36–ii41, 2003.

[38] Chandonia, J. M. and Karplus, M., "Neural networks for secondary structure and structural class predictions," *Protein Sci.*, vol. 4, pp. 275–285, 1995.

[39] Cheng, J. and Baldi, P., "Three-stage prediction of protein β-sheets by neural networks, alignments and graph algorithms," *Bioinformatics*, vol. 21 (Suppl. 1), pp. i75–i84, 2005.

[40] Cheng, J. and Baldi, P., "Improved residue contact prediction usign support vector machines and a large feature set," *BMC Bioinformatics*, vol. 8, no. 113, 2007.

[41] Cheng, J., Randall, A., Sweredoski, M., and Baldi, P., "SCRATCH: A protein structure and structural feature prediction server," *Nucleic Acids Res.*, vol. 33, pp. w72–w76, 2005.

[42] Chou, P. and Fasman, G., "Empirical predictions of protein conformation," *Annu. Rev. Biochem.*, vol. 47, pp. 251–276, 1978.

[43] Chou, P. Y. and Fasman, G. D., "Prediction of the secondary structure of the proteins from their amino acid sequence," *Adv. Enzymol. Relat. Areas Mol. Biol.*, vol. 47, pp. 45–148, 1978.

[44] Chu, W., Ghahramani, Z., Podtelezhnikov, A., and Wild, D. L., "Bayesian segmental models with multiple sequence alignment profiles for protein secondary structure and contact map prediction," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 3, no. 2, pp. 98–113, 2006.

[45] Chu, W., Ghahramani, Z., and Wild, D. L., "A graphical model for protein secondary structure prediction," in *International Conference on Machine Learning (ICML-04)*, pp. 161–168, 2004.

[46] Cochran, D. A. E. and Doig, A. J., "Effect of the N1 residue on the stability of the alpha-helix for all 20 amino acids," *Proein Sci.*, vol. 10, pp. 463–470, 2001.

[47] Cochran, D. A. E. and Doig, A. J., "Effect of the N2 residue on the stability of the alpha-helix for all 20 amino acids," *Proein Sci.*, vol. 10, pp. 1305–1311, 2001.

[48] Consortium, I. H. G. S., "Initial sequencing and analysis of the human genome," *Nature*, vol. 409, pp. 860–921, 2001.

[49] Cuff, J. A. and Barton, G. J., "Evaluation and improvement of multiple sequence methods for protein secondary structure prediction," *Proteins*, vol. 34, pp. 508–519, 1999.

[50] Cuff, J. A. and Barton, G. J., "Application of multiple sequence alignment profiles to improve protein secondary structure prediction," *Proteins: Struct. Funct. Genet.*, vol. 40, pp. 502–511, 2000.

[51] Dasgupta, S. and Bell, J. A., "Design of helix ends. amino acid preferences, hydrogen bonding and electrostatic interactions," *Int. J. Pept. Protein Res.*, vol. 41, pp. 499–511, 1993.

[52] Doig, A. J. and Baldwin, R. L., "N- and C-capping preferences for all 20 amino acids in alpha-helical peptides," *Protein Sci.*, vol. 4, pp. 1325–1336, 1995.

[53] Durbin, R., Eddy, S., Krogh, A., and Mitchison, G., *Biological Sequence Analysis: Probabilistic models of proteins and nucleic acids.* Cambridge University Press, 1981.

[54] Engel, D. E. and William, F. D., "Amino acid propensities are position-dependent throughout the length of α-helices," *J. Mol. Biol*, vol. 337, pp. 1195–1205, 2004.

[55] Fariselli, P., Martelli, P. L., and Casadio, R., "A new decoding algorithm for hidden markov models improves the prediction of topology of all-beta membrane proteins," *BMC Bioinformatics*, vol. 6 (Suppl. 4):S12, 2005.

[56] Fontana, P., Bindewald, E., Toppo, S., Velasco, R., Valle, G., and Tosatto, S. C. E., "The SSEA server for protein secondary structure alignment," *Bioinformatics*, vol. 21, no. 3, pp. 393–395, 2005.

[57] Francesco, V. D., Munson, P. J., and Garnier, J., "FORESST: Fold recognition from secondary structure prediction of proteins," *Bioinformatics*, vol. 15, no. 2, pp. 131–140, 1998.

[58] Frishman, D. and Argos, P., "Incorporation of non-local interactions in protein secondary structure prediction from the amino acid sequence," *Protein Eng.*, vol. 9, no. 2, pp. 133–142, 1996.

[59] Frishman, D. and Argos, P., "Seventy-five percent accuracy in protein secondary structure prediction," *Proteins*, vol. 27, pp. 329–335, 1997.

[60] Garnier, J., Gibrat, J., and Robson, B., "GOR method for predicting secondary structure from amino acid sequence," *Methods Enzymol.*, vol. 266, pp. 540–553, 1996.

[61] G.E. Crooks, J. W. and Brenner, S., "Measurements of protein sequence-structure correlations," *Proteins: Structure, Function, and Bioinformatics*, vol. 57, pp. 804–810, 2004.

[62] Geourjon, C. and Deleage, G., "SOPM: a self optimized method for protein secondary structure prediction," *Protein Eng.*, vol. 7, pp. 157–164, 1994.

[63] Gopalakrishnan, P. S., Bahl, L. R., and Mercer, R. L., "A tree-search strategy for large vocabulary continuous speech recognition," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, vol. 1, pp. 572–575, 1995.

[64] Gotoh, O., "An improved algorithm for matching biological sequences," *J. Mol. Biol.*, vol. 264, pp. 823–838, 1982.

[65] Guermeur, Y., Pollastri, G., Elisseeff, A., Zelus, D., Paugam-Moisy, H., and Baldi, P., "Combining protein secondary structure prediction models with ensemble methods of optimal complexity," *Neurocomputing*, vol. 56, pp. 305–327, 2003.

[66] Guo, J., Chen, H., Sun, Z., and Lin, Y., "A novel method for protein secondary structure prediction using dual-layer svm and profiles," *Proteins*, vol. 54, pp. 738–743, 2004.

[67] Hamilton, N., Burrage, K., Ragan, M., and Huber, T., "Protein contact prediction using patterns of correlation," *Proteins*, vol. 56, pp. 679–684, 2004.

[68] Henikoff, S. and Henikoff, J. G., "Amino acid substitution matrices from protein blocks," *P.N.A.S. USA*, vol. 89, pp. 10915–10919, 1992.

[69] Hobohm, U. and Sander, C., "Enlarged representative set of protein structures," *Protein Sci.*, vol. 3, pp. 522–524, 1994.

[70] Hou, Y., Hsu, W., Lee, M. L., and Bystroff, C., "Remote homology detection using local sequence-structure correlations," *Proteins: Structure, Function and Bioinformatics*, vol. 57, pp. 518–530, 2004.

[71] Hua, S. and Sun, Z., "A novel method of protein secondary structure prediction with high segment overlap measure: Support vector machine approach," *J. Mol. Biol.*, vol. 308, pp. 397–407, 2001.

[72] Huang, X. and Miller, W. A., "A time-efficient, linear-space local similarity algorithm," *Adv. Appl. Math*, vol. 12, pp. 337–357, 1991.

[73] Hubbard, T. J., "Use of β-strand interaction pseudo-potentials in protein structure prediction and modelling," in *Proceedings of the Biotechnology Computing Track Protein Structure Prediction MiniTrack of the 27th HICSS* (Lathrop, R. H., ed.), pp. 336–354, New York: IEEE Computer Society Press, 1994.

[74] Hubbard, T. J. and Park, J., "Fold recognition and ab initio structure predictions using hidden markov models and β-strand pair potentials," *Proteins: Struct. Func. Genet.*, vol. 23, pp. 398–402, 1995.

[75] Hutchinson, E. G., Sessions, R. B., Thornton, J. M., and Woolfson, D. N., "Determinants of strand register in antiparallel beta-sheets of proteins," *Protein Sci.*, vol. 7, pp. 287–300, 1998.

[76] Jelinek, F., "Fast sequential decoding algorithm using a stack," *IBM Journal of Research and Development*, vol. 13, pp. 675–685, 1969.

[77] Jensen, L. J., Gupta, R., Blom, N., Devos, D., Tamames, J., Kesmir, C., Nielsen, H., Staerfeldt, H. H., Rapacki, K., Workman, C., Andersen, C. A. F., Knudsen, S., Krogh, A., Valencia, A., and Brunak, S., "Prediction of human protein function from post-translational modifications and localization features," *J. Mol. Biol.*, vol. 319, pp. 1257–1265, 2002.

[78] Jensen, L. J., Skovgaard, M., Sicheritz-Ponten, T., Jorgensen, M. K., Lundegaard, C., Pedersen, C. C., Petersen, N., and Ussery, D., "Analysis of two large functionally uncharacterized regions in the methanopyrus kandleri AV19 genome," *BMC Genomics*, vol. 4, p. 12, 2003.

[79] Jeong, J., Berman, P., and Przytycka, T., "Bringing folding pathways into strand pairing prediction," in *The Workshop on Algorithms in Bioinformatics WABI*, vol. 4645, pp. 38–49, 2007.

[80] Jones, D. T., "Protein secondary structure prediction based on position-specific scoring matrices," *J. Mol. Biol.*, vol. 292, pp. 195–202, 1999.

[81] Kabsch, W. and Sander, C., "Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features," *Biopolymers*, vol. 22, pp. 2577–2637, 1983.

[82] Karplus, K., Barrett, C., and Hughey, R., "Hidden Markov models for detecting remote protein homologies," *Bioinformatics*, vol. I4, pp. 846–856, 1998.

[83] Kelley, L. A., MacCallum, R. M., and Sternberg, M. J. E., "Enhanced genome annotation using structural profiles in the program 3d-pssm," *J. Mol. Biol.*, vol. 299, pp. 501–522, 2000.

[84] Kim, H. and Park, H., "Protein secondary structure based on an improved support vector machines approach," *Protein Eng.*, vol. 16, pp. 553–560, 2003.

[85] Koh, E., Kim, T., and Cho, H. S., "Mean curvature as a major determinant of beta-sheet propensity," *Bioinformatics.*, vol. 22, pp. 297–302, 2006.

[86] Kortemme, T., Ramirez-Alvarado, M., and Serrano, L., "Design of a 20-amino acid, three-stranded β-sheet protein," *Science*, vol. 281, pp. 253–256, 1998.

[87] Krogh, A., "Two methods for improving performance of an HMM and their application for gene finding," *J. Mol. Biol.*, vol. 219, pp. 727–732, 1991.

[88] Kuhlman, B., Dantas, G., Ireton, G., Varani, G., Stoddard, B., and Baker, D., "Design of a novel globular protein fold with atomic-level accuracy," *Science*, vol. 302, pp. 1364–1368, 2003.

[89] Kulp, D., Haussler, D., Reese, M. G., and Eeckman, F. H., "A generalized hidden Markov model for the recognition of human genes in DNA," in *Proc. Int. Conf. Intell. Syst. Mol. Biol. (ISMB-96), St. Louis, MO*, vol. 4, pp. 134–142, AAAI/MIT Press, 1996.

[90] Kumar, S. and Bansal, M., "Dissecting alpha-helices: position specific analysis of alpha-helices in globular proteins," *Proteins*, vol. 31, pp. 460–476, 1998.

[91] Levin, J. M., "Exploring the limits of nearest neighbour secondary structure prediction," *Protein Eng.*, vol. 10, pp. 771–776, 1997.

[92] Lifson, S. and Sander, C., "Specific recognition in the tertiary structure of beta-sheets of proteins," *J. Mol. Biol.*, vol. 139, pp. 627–639, 1980.

[93] MacCallum, R., "Striped sheets and protein contact prediction," *Bioinformatics*, vol. 20 (Supplement 1), pp. i224–i231, 2004.

[94] Mandel-Gutfreund, Y., Zaremba, S. M., and Gregoret, L. M., "Contributions of residue pairing to beta-sheet formation: conservation and covariation of amino acid residue pairs on antiparallel beta-strands," *J. Mol. Biol.*, vol. 305, pp. 1145–1159, 2001.

[95] Matthews, B. W., "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochim. Biophys. Acta.*, vol. 405, pp. 442–451, 1975.

[96] Meiler, J., Mueller, M., Zeidler, A., and Schmaeschke, F., "Generation and evaluation of dimension-reduced amino acid parameter representations by artificial neural networks," *J. Mol. Model.*, vol. 7, pp. 360–369, 2001.

[97] Merkel, J. S. and Regan, L., "Modulating protein folding rates in vivo and in vitro by side-chain interactions between the parallel beta strands of green fluorescent protein," *J. Biol. Chem.*, vol. 275, pp. 29200–29206, 2000.

[98] Minor, D. L. and Kim, S., "Context is a major determinant of beta-sheet propensity," *Nature*, vol. 371, pp. 264–267, 1994.

[99] Mirny, L. A. and Shakhnovich, E. I., "Protein structure prediction by threading. why it works and why it does not," *J Mol. Biol.*, vol. 283, pp. 507–526, 1998.

[100] Montelione, G. T. and Anderson, S., "Structural genomics: keystone for a Human Proteome Project," *Nature Struct. Biol.*, vol. 6, pp. 11–612, 1999.

[101] Moult, J., Fidelis, K., Zemla, A., and Hubbard, T., "Critical assessment of methods of protein structure prediction (CASP): round iv," *Proteins*, vol. 45, no. (Suppl. 5), pp. 2–7, 2001.

[102] M.S. Cline, K. Karplus, R. L. T. S. R. R. J. and Haussler, D., "Information-theoretic dissection of pairwise contact potentials," *Proteins: Structure, Function, and Bioinformatics*, vol. 49, pp. 7–14, 2002.

[103] Needleman, S. B. and Wunsch, C. D., "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *J. Mol. Biol.*, vol. 48, pp. 443–453, 1970.

[104] Nguyen, M. N. and Rajapakse, J. C., "Multi-class support vector machines for protein secondary structure prediction," *Genome Inform.*, vol. 14, pp. 218–227, 2003.

[105] Nguyen, M. N. and Rajapakse, J. C., "Two-stage support vector machines for protein secondary structure prediction," *Neu. Par. Sci. Comp.*, vol. 11, pp. 1–18, 2003.

[106] Nilsson, D. and Goldberger, J., "Sequentially finding the N-Best list in hidden Markov models," *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, 2001.

[107] Nilsson, N. J., *Problem Solving Methods of Artificial Intelligence*. New York: McGraw-Hill, 1971.

[108] Park, J., Teichmann, S. A., Hubbard, T., and Chothia, C., "Intermediate sequences increase the detection of distant sequence homologies," *J. Mol. Biol.*, vol. 273, pp. 349–354, 1997.

[109] Parker, L., "Cs302 lecture notes: Topological sort/cycle detection." http://www.cs.utk.edu/∼parker/Courses/CS302-fall03/Notes/GraphIntro/.

[110] Paul, D. B., "An efficient A* stack decoder algorithm for continuous speech recognition with a stochastic language model," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Proc. (ICASSP'92)*, vol. 1, pp. 25–28, 1992.

[111] Penel, S., Morrison, R. G., Mortishire-Smith, R. J., and Doig, A. J., "Periodicity in alpha-helix lengths and C-capping preferences," *J. Mol. Biol.*, vol. 293, pp. 1211–1219, 1999.

[112] Petersen, T. N., Lundegaard, C., Nielsen, M., Bohr, H., Bohr, J., Brunak, S., Gippert, G. P., and Lund, O., "Prediction of protein secondary structure at 80% accuracy," *Proteins*, vol. 41, pp. 17–20, 2000.

[113] Pollastri, G. and Baldi, P., "Prediction of contact maps by GIOHMMs and recurrent neural networks using lateral propagation from all four cardinal corners," *Bioinformatics*, vol. I8 (Suppl. I), pp. S62–S70, 2002.

[114] Pollastri, G. and McLysaght., A., "Porter: a new, accurate server for protein secondary structure prediction," *Bioinformatics*, vol. 21, pp. 1719–20, 2005.

[115] Pollastri, G., Przybylski, D., Rost, B., and Baldi, P., "Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles," *Proteins*, vol. 47, pp. 228–235, 2002.

[116] Presta, L. G. and Rose, G. D., "Helix signals in proteins," *Science*, vol. 240, pp. 1632–1641, 1988.

[117] Przybylski, D. and Rost, B., "Alignments grow, secondary structure prediction improves," *Proteins*, vol. 46, pp. 197–205, 2002.

[118] Punta, M. and Rost, B., "PROFcon: novel prediction of long-range contacts," *Bioinformatics*, vol. 21, pp. 2960–2968, 2005.

[119] Rabiner, L. R., "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[120] Raghava, G. P. S., "APSSP2: Protein secondary structure prediction using nearest neighbor and neural network approach," *CASP4: 75–76*, 2000.

[121] Randall, A., Cheng, J., Sweredoski, M., and Baldi, P., "TMBpro: secondary structure, β-contact and tertiary structure prediction of transmembrane β-barrel proteins," *Bioinformatics*, vol. 24, pp. 513–520, 2008.

[122] Richardson, J. S. and Richardson, D. C., "Amino acid preferences for specific locations at the ends of alpha helices," *Science*, vol. 240, pp. 1648–1652, 1988.

[123] Robles, V., Larrañaga, P., Peña, J., Menasalvas, E., Pérez, M., and Herves, V., "Bayesian networks as consensed voting system in the construction of a multi-classifier for protein secondary structure prediction.," *Artificial Intelligence in Medicine, special issue in "Data mining in genomics and proteomics"*, vol. 31, pp. 117–136, 2004.

[124] Rost, B., "Twilight zone of protein sequence alignments," *Protein Eng.*, vol. 12, pp. 85–94, 1999.

[125] Rost, B., "Rising accuracy of protein secondary structure prediction," in *Protein structure determination, analysis, and modeling for drug discovery* (Chasman, D., ed.), pp. 207–249, New York: Dekker, 2003.

[126] Rost, B. and Eyrich, V. A., "EVA: large-scale analysis of secondary structure prediction," *Proteins*, vol. 45, no. (Suppl. 5), pp. 192–199, 2001.

[127] Rost, B., Liu, J., Przybylski, D., Nair, R., Wrzeszczynski, K., Bigelow, H., and Ofran, Y., "Prediction of protein structure through evolution," in *Handbook of ChemoinformaticsFrom Data to Knowledge.* (Gasteiger, J. and Engel, T., eds.), pp. 1789–1811, New York: Wiley, 2003.

[128] Rost, B. and Sander, C., "Prediction of protein secondary structure at better than 70 accuracy," *J. Mol. Biol.*, vol. 232, pp. 584–599, 1993.

[129] Rost, B., Sander, C., and Schneider, R., "Redefining the goals of protein secondary structure prediction," *J. Mol. Biol.*, vol. 235, pp. 13–26, 1994.

[130] Ruczinski, I., *Logic Regression and Statistical Issues Related to the Protein Folding Problem.* PhD thesis, Department of Statistics, University of Washington, Seattle, WA, 2000.

[131] Ruczinski, I., Kooperberg, C., Bonneau, R., and Baker, D., "Distributions of beta sheets in proteins with application to structure prediction," *Proteins: Structure, Function and Genetics*, vol. 48, pp. 85–97, 2002.

[132] Salamov, A. A. and Solovyev, V. V., "Protein secondary structure prediction using local alignments," *J. Mol. Biol.*, vol. 268, pp. 31–36, 1997.

[133] Saqi, M. A. S. and Sternberg, M. J. E., "A simple method to generate non-trivial alternate alignments of protein sequences," *J. Mol. Biol.*, vol. 219, pp. 727–732, 1991.

[134] Schmidler, S. C., Liu, J. S., and Brutlag, D. L., "Bayesian segmentation of protein secondary structure," *J. Comp. Biol.*, vol. 7, pp. 233–248, 2000.

[135] Schmidler, S. C., Liu, J. S., and Brutlag, D. L., "Bayesian protein structure prediction," *Case Studies in Bayesian Statistics*, vol. 5, pp. 363–378, 2001.

[136] Schwartz, R. and Austin, S., "A comparison of several approximate algorithms for finding multiple (N-Best) sentence hypothesis," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Proc. (ICASSP'91)*, vol. 1, pp. 701–704, 1991.

[137] Schwartz, R. and Chow, Y. L., "The N-Best algorithm: An efficient and exact procedure for finding the N most likely sentence hypothesis," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Proc. (ICASSP'90)*, vol. 1, pp. 81–84, 1990.

[138] Smith, C. K. and Regan, L., "Guidelines for protein design: The energetics of β sheet side chain interactions," *Science*, vol. 270, pp. 980–982, 1995.

[139] Soong, F. K. and Huang, E. F., "A tree-trellis based fast search for finding the N Best sentence hypotheses in continuous speech recognition," in *Proceedings of a workshop on Speech and natural language*, pp. 12–19, 1990.

[140] Steward, R. E. and Thornton, J. M., "Prediction of strand pairing in antiparallel and parallel beta-sheets using information theory," *Proteins Struct. Funct. Genet.*, vol. 48, pp. 178–191, 2002.

[141] Tsigelny, F. I., *Protein Structure Prediction: Bioinformatic Approach*. International University Lane, 2002.

[142] Vullo, A., Walsh, I., and Pollastri, G., "A two-stage approach for improved prediction of residue contact maps," *BMC Bioinformatics*, vol. 7, no. 180, 2006.

[143] Waldispuhl, J., Berger, B., Clote, P., and Steyaert, J. M., "Predicting transmembrane β-barrels and interstrand residue interactions from sequence," *PROTEINS: Structure, Function, and Bioinformatics*, vol. 65, pp. 61–74, 2006.

[144] Wallqvist, A., Fukunushi, Y., Murphy, L. R., Fadel, A., and Levy, R. M., "Iterative sequence/secondary structure search for protein homologs: comparison with amino acid sequence alignments and application to fold recognition in genome databases," *Bioinformatics*, vol. 16, no. 11, pp. 988–1002, 2000.

[145] Wang, K. and Samudrala, R., "FSSA: A novel method for identifying functional signatures from structural alignments," *Bioinformatics*, vol. 21, no. 13, pp. 2969–2977, 2005.

[146] Ward, J. J., McGuffin, L. J., Buxton, B. F., and Jones, D. T., "Secondary structure prediction with support vector machines," *Bioinformatics*, vol. 19, pp. 1650–1655, 2003.

[147] Waterman, M. S. and Eggert, M., "A new algorithm for best subsequence alignments with application to tRNA-rRNA comparisons," *J. Mol. Biol.*, vol. 197, pp. 723–725, 1987.

[148] Woolfson, D. N., Evans, P. A., Hutchinson, E. G., and Thornton, J. M., "On the conformation of proteins: The handedness of the connection between parallel β-strands," *J. Mol. Biol.*, vol. 110, pp. 269–283, 1977.

[149] Wouters, M. A. and Curmi, P. M. G., "An analysis of side chain interactions and pair correlations within antiparallel beta-sheets: the differences between backbone hydrogen bonded and non-hydrogen bonded residue pairs," *Proteins Struct. Func. Genet.*, vol. 22, pp. 119–131, 1995.

[150] Zaremba, S. M. and Gregoret, L. M., "Context-dependence of amino acid residue pairing in antiparallel β-sheets," *J. Mol. Biol.*, vol. 291, pp. 463–479, 1999.

[151] Zemla, A., Venclovas, C., Fidelis, K., and Rost, B., "A modified definition of SOV, a segment-based measure for protein secondary structure prediction assessment," *Proteins*, vol. 34, pp. 220–223, 1999.

[152] Zhang, C. and Kim, S., "The anatomy of protein beta-sheet topology," *J. Mol. Biol.*, vol. 2, pp. 1075–1089, 2000.

[153] Zhu, H. and Braun, W., "Sequence specificity, statistical potentials, and three-dimensional structure prediction with selfcorrecting," *Protein Sci.*, vol. 8, pp. 326–342, 1999.

VITA

Zafer Aydın received the B.S. and M.S. degrees in Electrical Engineering from Bilkent University, Ankara, Turkey, in 1999, and 2001, respectively. He received his Ph.D. degree in Electrical Engineering from Georgia Institute of Technology in 2008. His research interests include bioinformatics, computational biology, pattern recognition, and machine learning.