Unravelling the Architecture of Membrane Proteins with Conditional Random Fields

Lior Y. Lukov

Master of Science by Research

School of Information Technologies

Supervisor: Dr Sanjay Chawla

The University of Sydney

August, 2005

Abstract

In this thesis we use Conditional Random Fields (CRFs) as a sequential classifier to predict the location of transmembrane helical regions in membrane proteins. CRFs allow for a seamless and principled integration of biological domain knowledge into the model and are known to have several advantages over other approaches. We have used this flexibility in order to incorporate several biologically inspired features into the model. We compared our approach with twenty eight other methods and received the highest score in the percentage of residues predicted correctly. We have also carried out experiments comparing CRFs against Maximum Entropy Models (MEMMs). Our results confirm that CRFs overcome the label bias problem, which are known to afflict MEMMs. Furthermore, we have used CRFs to analyze the architecture of the protein complex, Cytochrome c oxidase, and have recreated the results obtained from physical experiments.

Acknowledgements

Many thanks to my supervisor, Sanjay Chawla, for his tremendous job guiding me during this thesis, for spending so many hours, giving the right advice at the right time and for his outstanding contribution to my studies.
Also many thanks to W. Bret Church for his continuous care, good advice and willingness to help, for sharing his extensive biology knowledge and for making me welcome in his lab.
Development of our model was based on the JAVA CRFs implementation package of Conditional Random Fields for sequential labelling, developed by Sunita Sarawagi of IIT Bombay. We would like to thank Sunita Sarawagi for sharing her work with the research community, sourceforge.net for hosting and distributing this useful information, Alex Smola for his helpful suggestions on CRFs, and Tara McIntosh for her useful comments.

The experiments on Maximum Entropy were based on the JAVA MaxEnt implementation package, developed by Dan Klein, and we would like to thank him for sharing his code.

Finally, I would like to thank my family who supported me so much during my whole period of studies.

Contents

1 Introduction **1**

1.1 Background 1

1.2 Problem Definition 2

1.3 Key Contributions 4

1.4 CRFs and MEMMs Model Implementation 5

1.5 Organization of this Thesis 5

2 Proteins and Proteomics **7**

2.1 Amino Acids 8

2.2 Introduction to Proteins 11

2.2.1 Homologous Proteins 11

2.2.2 Proteomics 12

2.2.3 Protein Structure 13

2.2.4 Integral Membrane Proteins (IMPs) 17

2.3 Structure Prediction Using Bioinformatics 18

2.3.1 Identifying Protein Domains 18

2.3.2 Secondary Structure Prediction 20

2.3.3 Tertiary Structure Prediction 21

2.3.4 Transmembrane Segments Prediction 21

2.4 Summary 22

3 The Sequential Classification Problem **24**

3.1 Directed Graphical Models 25

3.2 Hidden Markov Models (HMMs) 26
3.2.1 Sequential Classification Example with HMMs 29
3.3 Generative and Conditional Models 32
3.4 Maximum Entropy Markov Models (MEMMs) 33
3.5 MEMMs Label Bias Problem 35
3.6 Summary 38

4 Conditional Random Fields (CRFs) 40
4.1 Undirected Graphical Models 41
4.2 CRF Definition and Model Derivation 42
4.3 Feature Functions and Model Estimation 43
4.4 The Maximum Likelihood Approach 45
4.5 Parameter Estimation 47
4.5.1 Iterative Scaling 48
4.5.2 Generalized Iterative Scaling (GIS) Algorithm 49
4.5.3 Improved Iterative Scaling (IIS) Algorithm 51
4.5.4 Newton's Method 52
4.5.5 Quasi-Newton Methods 55
4.5.6 Limited Memory BFGS Method 56
4.6 Conditional Probability in Matrix Form 57
4.7 Feature Integration with the Model 59
4.7.1 Start, End and Edge Features 60
4.7.2 Basic Amino Acid Features 61
4.7.3 Single Side Neighboring Amino Acid Features 61
4.7.4 Single Side Shuffled Neighboring Amino Acid Features . 62
4.7.5 Double Side Neighboring Amino Acid Features 62
4.7.6 Double Side Shuffled Neighboring Amino Acid Features . 63
4.7.7 Amino Acid Property Features 64
4.7.8 Border Features 65
4.8 CRFs Prediction Example 67
4.9 Summary 70

5 Experiments, Evaluation and Analysis **71**

5.1 Data Set 72

5.2 Prediction Metrics 74

5.2.1 Per-Residue Accuracy 74

5.2.2 Per-Segment Accuracy 74

5.3 Results and Analysis 76

5.3.1 Transmembrane Helix Prediction 76

5.3.2 Local Residue Distribution in Transmembrane Regions . . 80

5.3.3 Comparison of CRFs and MEMMs 81

5.3.4 Cytochrome c oxidase Protein Analysis 87

5.3.4.1 Approach 92

5.4 Summary 94

5.5 Conclusions 95

Bibliography **96**

List of Figures

1.1 Examples of IMPs sequence pairs 3

2.1 Amino acids reaction forming a peptide bond 10
2.2 Polypeptide backbone 10
2.3 Polypeptide backbone with side chains 10
2.4 Full polypeptide chain 10
2.5 α-helix structure representation 15
2.6 Parallel β-sheets structure representation 16
2.7 Antiparallel β-sheets structure representation 16
2.8 Primary, secondary, tertiary and quaternary structures 17
2.9 Transport of molecules through IMPs 20

3.1 Labelling protein secondary structure 25
3.2 Hidden Markov models graphical structure 27
3.3 Maximum entropy Markov models graphical structure 34
3.4 Label bias problem example 36

4.1 CRFs graphical structure 41
4.2 Geometric interpretation of Newton's method 54

5.1 Transmembrane helix prediction experimental flow 73
5.2 Per-segment accuracy example 75
5.3 Prediction score for selected feature combinations 77
5.4 CRFs vs. MEMMs performance test on Experiment 2 82

5.5 CRFs vs. MEMMs performance test on Experiment 3 82
5.6 CRFs vs. MEMMs performance test on Experiment 4 83
5.7 CRFs vs. MEMMs performance test on Experiment 5 83
5.8 Conditional Probability on Different Transition States 85
5.9 CRFs vs. MEMMs prediction output 86
5.10 Crystal structure of Cytochrome c oxidase 88
5.11 Predicted vs. observed residue frequencies in the membrane . . . 89
5.12 Properties of the predicted residues in the membrane 89
5.13 Frequency of leucine in the membrane 90
5.14 Frequency of isoleucine in the membrane 90
5.15 Frequency of valine in the membrane 91
5.16 Frequency of arginine in the membrane 91

List of Tables

2.1 Amino acids ordered alphabetically. 8
2.2 Set of websites for identifying protein structure 19

3.1 Conditional probability of amino acids in transmembrane helix . . 31
3.2 Conditional probability of transition between helical states 31
3.3 Maximizing HMMs joint distribution using dynamic programming 32
3.4 Solving the maximization problem using dynamic programming . 32

4.1 The list of all features used 66
4.2 Activated features on a training set example 68
4.3 Trained feature parameters 69
4.4 Sequence label calculation 69

5.1 The five metrics used to measure per-residue accuracy 76
5.2 The three metrics used for measuring per-segment accuracy 76
5.3 Enabled and disabled feature combination 77
5.4 Prediction score comparison between 29 methods 79
5.5 Local residue distribution and TMH prediction 81
5.6 Prediction results using different combinations 81
5.7 Prediction score comparison between CRFs and MEMMs 84

Chapter 1

Introduction

1.1 Background

An extremely abundant and important class of non-standard proteins are the Integral Membrane Proteins (IMPs) which are found permanently integrated in the cell membrane. IMPs control a broad range of events essential to the proper functioning of cells, tissues and organisms. These events include (i) the regulation of ion and metabolite fluxes, (ii) maintaining appropriate contact between cells, connective tissues, and extracellular matrices, and (iii) transducing extracellular signals to intracellular signalling pathways [6]. In addition IMPs include families that are the most common target of prescription drugs [48].

A number of high throughput projects have been positioned to assist in the interpretation of the human genome sequence data, as well as sequence data from other key organisms. Among such efforts are the International SNP (single-nucleotide polymorphism) consortium and various US National Structural Genomics efforts [1]. The first crystal structure for a protein of unknown function was reported in early 2000 [49] but only one of the many structural genomics programs that are now underway has mentioned membrane proteins, suggesting a valuable role for integrated approaches to the problem of solving the structures of IMPs. The al-

gorithms that assess sequences and structural data have been applied in assessing data generated from these projects.

Despite both their abundance and importance, the structural information available for IMPs is in a limited state as very few high-resolution 3D structures are available. The total number of IMPs analysed is less than 100 (at a resolution of 4Å or better), while the complete protein collection at the Research Collaboratory for Structural Bioinformatics data bank is now approximately 30,000 [3].

In order to determine the function of a protein it is necessary to know its structure. However, for IMPs this task turns out to be difficult. The shortage of data on the structures of membrane proteins from traditional biophysical methods in comparison to water-soluble proteins stems from a lack of success of X-ray crystallography and NMR spectroscopy on these proteins. Structural determination of integral membrane proteins can be problematic due to difficulties in obtaining sufficient amounts of sample. A further contributing factor is that IMPs are generally much more likely be become inactive while handling or waiting to crystallize. Mass spectroscopy has begun to assist topology assignment in rapid product identification in limited proteolytic cleavage experiments [22]. This is an area where data driven methods may be suitable to contribute significantly to the study of structure and function. Therefore good prediction methods for IMPs structures are highly valued.

1.2 Problem Definition

In this thesis we cast the transmembrane helix prediction task as a binary sequential classification problem and use Conditional Random fields (CRFs) to solve it [21].

CRFs are a probabilistic classification framework for labelling sequential data. Given a set of membrane protein sequences, each single record in the set contains a pair of sequences: The observation sequence represented by x, and the label

sequence represented by y. The protein observation sequence consists of amino acids, represented by 20 different letters. The second sequence in the pair consists of a transmembrane helical/non-helical structure, and is also known as the label sequence. Each letter in this sequence corresponds to the transmembrane helical/non-helical state of the protein in that location, and is represented by binary labels 0/1 (0= non helical structure, 1= helical structure). A label at position i corresponds to the amino acids's helical structure at the same position. An example set of sequence pairs is shown in Figure 1.1.

```
MFINRWLFSTNHKDIGTLYLLFGAWAGMVGTALSLLIRAELGQPGTLLGDDQIYNVVVTAHAFV
OOOOOOOOOOO1111111111111111111111111111OOOOOOOOOO11111111111111

MAYPMQLGFQDATSPIMEELLHFHDHTLMIVFLISSLVLYIISLMLTTKLTHTSTMDAQEVETI
OOOOOOOOOOOOOOOOOO1111111111111111111111OOOOOOOOOOOOOOOOOO11111

MTHQTHAYHMVNPSPWPLTGALSALLMTSGLTMWFHFNSMTLLMIGLTTNMLTMYQWWRDVIRE
OOOOOOOOOOOOOOO1111111111111111111OOOOOO11111111111111111111111

MENLNMDLLYMAAAVMMGLAAIGAAIGIGILGGKFLEGAARQPDLIPLLRTQFFIVMGLVDAIP
OOOOOOOOOO1111111111111111111OOOOOOOOOOOOOOOOOOOOO111111111111

MNGTEGPNFYVPFSNKTGVVRSPFEAPQYYLAEPWQFSMLAAYMFLLIMLGFPINFLTLYVTVQ
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO1111111111111111111111111O
```

Figure 1.1: Examples of IMPs sequence pairs.

Given a set S of observation sequences with unknown labels, the objective of the classification problem is to build a model using a set of training sequences with known labels, and then to predict the labels of the observation sequences of S.

CRFs allow for a modular integration of biological domain knowledge expressed as model constraints. This knowledge is incorporated into the model by selecting a set of features that capture the relationships between the observation and the label sequences, in our case the protein sequence and the helical structure respectively.

1.3 Key Contributions

In this thesis we made several contributions, presented in the following order:

1. Conditional Random Fields for transmembrane helix prediction:
   We have tested the performance of CRFs to predict the location of membrane helical regions in protein sequences and compared our results with other available methods. The CRFs model achieved the highest score of 83% among all the twenty nine methods in the percentage of residues correctly predicted. On segment prediction, CRFs achieved the fourth highest score. For 75% of the proteins all the transmembrane helices were correctly predicted. The total percentage of correctly predicted helices was 94% with a precision of 91%. We have also shown how CRFs allow for a seamless and principled integration of biological domain knowledge into the model and presented several advantages of CRFs over other approaches.

2. CRFs vs MEMMs performance comparison on transmembrane prediction:
   We have carried out a detailed comparison between MEMMs and CRFs and shown that CRFs outperform MEMMs on several important measures. Furthermore, we have also provided a detailed analysis of the label bias problem that MEMMs suffer from.

3. Cytochrome c oxidase analysis:
   We have used CRFs to analyze the architecture of the protein complex, Cytochrome c oxidase. Cytochrome c oxidase, as a single entity provides many helices for examination - it is a complex of 13 subunits, 10 of which provide 28 entire transmembrane passes and is one of the most well studied complexes [40]. One of our key contributions is that we were able to demonstrate and predict the propensities of amino acids to exist in regions of the membrane using our prediction methods as was reported by Wallin et al. [46].

4. Feature selection:
   We have carried out an extensive test to select the most appropriate features to embed in the CRFs model. A detailed evaluation is presented in this thesis.

5. Novel approach using a known but new technique:
   CRFs were first introduced by Lafferty et al. on 2001 in the 18th International Conference on Machine Learning in California. At the time when the research commenced in early 2004, most of the applications of CRFs were in the natural language processing domain. The literature was lacking applications of CRFs in bioinformatics domain, and our work was novel [26]. Today, the use of CRFs in bioinformatics is growing and successful applications, such as Gene prediction using CRFs, have appeared in the literature [7].

1.4 CRFs and MEMMs Model Implementation

Development of our model was based on the JAVA CRFs implementation package of Conditional Random Fields for sequential labelling, developed by Sunita Sarawagi of IIT Bombay [36]. The experiments on Maximum Entropy were based on the JAVA MaxEnt implementation package, developed by Dan Klein [19].

1.5 Organization of this Thesis

The rest of this thesis is as follows: Chapter 2 gives a short introduction to proteins and the transmembrane proteins family. In Chapter 3 we describe the abstract sequential classification problem. Chapter 4 introduces the Conditional Random Fields approach, discusses it's main advantages, and gives a detailed explanation about the features used within the model. Chapter 5 presents experimental setup

and evaluation of our results on a benchmark data set, an empirical comparison between CRFs and MEMMs, and a detailed validation on the Cytochrome c oxidase complex. We conclude with a brief summary and directions for future research.

Chapter 2

Proteins and Proteomics

In this Chapter our objective is to provide a self-contained introduction to the biological aspects of protein structure and functionality. Proteins play a central role in all aspects of cell structure and function. Almost all biological reactions are involved with the functionality of protein molecules. Proteins are responsible for controlling cellular metabolism, producing other proteins, regulating the movement of molecular and ionic species across membranes, storing or converting cellular energy, and numerous other functions. The way each protein is generated, folded into a unique structure and inherits its designated functionality, resides in the genetic information of the cell. This information is coded in a sequence of nucleotide bases, known as the DNA (Deoxyribonucleic acid) sequence. The DNA sequence consists of thousands of segments of encoded information which define the genes. The information encapsulated in the genes is the key for protein formation. During a gene's expression the protein is synthesized according to the instructions encoded in the gene. The proteins are said to be the agents of biological function [11].

2.1 Amino Acids

All proteins found in nature are composed of 20 common building blocks known as amino acids. The protein's biological function is directly derived from this composition, as each amino acid has unique properties. All amino acids share a tetrahedral structure (except proline), having an alpha carbon (C_α), which is covalently linked to both the amino group (NH_3) and the carboxyl group (COO). This carbon is also bonded to a hydrogen and to a variable side chain group (R). In neutral solution the carboxyl group exists as -COO^- and the amino group as -NH_3^+ [11]. The side chain distinguishes one amino acid from another.

All amino acids are represented by a three letter code or one letter code, e.g: Leu or L respectively, for the amino acid Leucine. Table 2.1 lists the 20 amino acids with their name, three letter code and one letter code. In this thesis, when we deal with protein sequences, we adopt the one letter code for the amino acids forming the sequence.

Table 2.1: Amino acids ordered alphabetically.

Letter	Name	Abbrevation	Letter	Name	Abbrevation
A	Alanine	Ala	M	Methionine	Met
C	Cysteine	Cys	N	Asparagine	Apn
D	Aspartic Acid	Asp	P	Proline	Pro
E	Glutamic Acid	Glu	Q	Glutamine	Gln
F	Phenylalanine	Phe	R	Arginine	Arg
G	Glycine	Gly	S	Serine	Ser
H	Histidine	His	T	Threomine	Thr
I	Isoleucine	Ile	V	Valine	Val
K	Lysine	Lys	W	Tryptophan	Trp
L	Leucine	Leu	Y	Tyrosine	Tyr

There are different ways to classify the amino acids based on their properties. One common classification is the one defined by Sternberg [39], classifying the amino acids into nine groups:

1. Aromatic (F,W,Y,H)
2. Hydrophobic (M,I,L,V,A,G,F,W,Y,H,K,C)
3. Positive (H,K,R)
4. Polar (W,Y,C,H,K,R,E,D,S,Q,N,T)
5. Charged (H,K,R,E,D)
6. Negative (E,D)
7. Aliphatic (I,L,V)
8. Small (V,A,G,C,P,S,D,T,N)
9. Tiny (A,G,S)

This classification will be used in Section 5.3.3 to define features for the task of secondary structure prediction of proteins.
The amino group and the carboxyl group of the amino acids provide them with the ability to polymerize and form polypeptides and proteins. When two amino acids appear together, the amino group of the one and the carboxyl group of the other react together, forming a peptide bond and eliminating a water molecule as shown in Figure 2.1. Several iterations of this reaction (between different combinations of amino acids) produce polypeptides and proteins as shown in Figure 2.4.

Peptide describes short polymers of amino acids. A peptide is classified by the number of amino acids forming the chain. Each unit in the chain is called an amino acid residue. When the number of amino acids in the chain exceeds several dozen then the peptide is named a polypeptide. The length of polypeptide chains of proteins may run into the thousands. The average polypeptide length in the

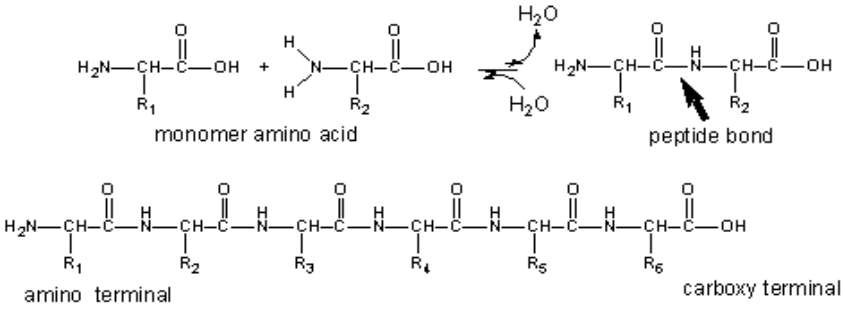


Figure 2.1: Amino acids reaction forming a peptide bond [15].

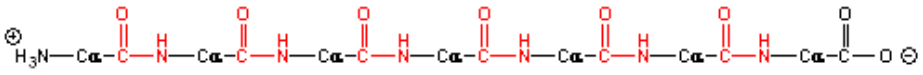


Figure 2.2: Polypeptide backbone [15].

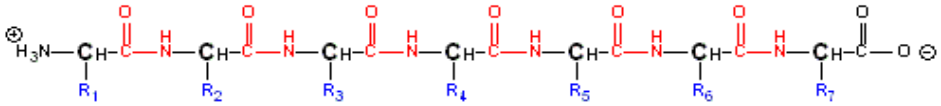


Figure 2.3: Polypeptide backbone together with amino acids side chains [15].

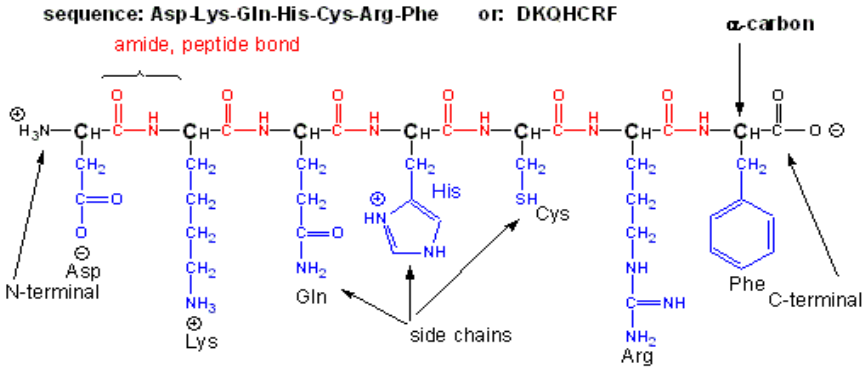


Figure 2.4: Full polypeptide chain of a protein with the amino acid sequence: Asp-Lys-Gln-His-Cys-Arg-Phe [15].

eukaryotic[1]kingdom is about 270 residues, with a molecular weight of 32000 daltons. There is no known correlation between peptide length and protein function, as proteins with different length may have similar functionality [11].

2.2 Introduction to Proteins

Proteins molecules are composed of one or more polypeptide chains. Chemically, proteins are chains of amino acids linked head to tail, starting from the N-terminus of the amino group to the C-terminus of the carboxyl group, forming covalent peptide bonds. The peptide backbone of a protein consists of the repeated sequence of $-N - C_\alpha - C-$ as shown in Figure 2.2.
C_α is the amino acid $\alpha-$carbon which is attached to a side chain R. The second carbon is attached to oxygen and to the nitrogen N of the following amino acid. Figure 2.3 illustrates the general polypeptide structure. The carbonyl oxygen and the amide hydrogen are energetically favorable to appear diagonally in the chemical *trans* conformation.

The unique characteristic of each protein is directly derived from the sequence of amino acid residues in its polypeptide chain. By convention, the amino acid sequence is read from the N-terminal end of the polypeptide chain through to the C-terminal end. Given that for every position in the sequence there are 20 possible amino acids, the number of unique amino acid sequences is astronomical, even for a protein with tens of residues [23].

2.2.1 Homologous Proteins

Proteins having a significant degree of similarity in their amino acid sequence are called *homologous proteins*. Homologous proteins are also proteins in different

[1]A single-celled or multicellular organism whose cells contain a distinct membrane-bound nucleus.

organisms which perform the same functionality. Having high similarity among homologous proteins from different species indicates common ancestry. If we look, for example, at the sequence of Cytochrome c oxidase protein found in the mitochondria of all eukaryotic organisms, it contains a few more than 100 amino acids. This protein has 28 positions in its sequence where the same amino acids are always to be found among 40 different species. This suggests that these positions are crucial to the biological functionality of this protein. Species which are more close to each other share higher sequence similarities. Humans and chimpanzees for example, share an identical Cytochrome c oxidase protein sequence, while the sequence among humans and other mammals differ at 10 residues, and among humans and reptiles at 14 residues [23].

2.2.2 Proteomics

The term Proteome was coined to describe the complete set of proteins that is expressed by the entire genome in the lifetime of a cell. Functional Proteomics is the study of the functionality of all proteins. It includes the study of expressed proteins of a genome using two dimensional (2D) gel electrophoresis, and mass spectrometry (MS) to separate and identify proteins, and the study of protein-protein interactions [47].

The principal of mass spectrometry is to separate ionized atoms or molecules from each other based on their difference in mass to charge (m/z) ratio. The mass to charge ratio is a highly characteristic property for determination of chemical and structural information of molecules. To accomplish this task, the mass spectrometer is used to evaporate and ionize molecules in a vacuum to create a gas of ions. The ions are separated based on their mass to charge ratio and the amount of ions with specific mass to charge ratio are measured. Mass spectrometry for proteins has to be modified because they decompose by heat rather than evaporation. Such techniques include Electrospray ionization (ESI-MS), Fast atom bombardment (FAB-MS), Laser ionization (LIMS), and Matrix assisted desorption ion-

ization (MALDI). Eventually, by using one of these techniques, the spectrometer generates a unique spectrum of mass to charge ratio of the ionized protein having a peak at the correct protein mass. Finally, the protein is sequenced by a Tandem mass spectrometer (Tandem MS or MS/MS). The Tandem MS is actually made of two mass spectrometers connected in a row. The first spectrometer separates the protein from its different ionized peptides, which are transferred to the second spectrometer. Here it is fragmented by collision with helium or argon gas molecules to several fragments which are then analyzed. The fragmentation occurs in the peptide bonds linking two amino acids in a peptide chain, and creates new set of peptides differing in length by one amino acid. The sequences retrieved from different fragments are aligned based on sequence overlaps which are finally composed to an overall amino acid sequence of the polypeptide [11].

A different approach to reveal the protein's sequence information, is from translating the nucleotide sequences of genes into triplets of nucleotides, known as codons, and from codons into amino acid sequences. This methods is faster, more accurate and more efficient than revealing the sequence directly from a protein.

2.2.3 Protein Structure

Proteins are polymers of amino acids containing a constant main chain of repeating units, the backbone, and variable side chains of the different amino acids. The amino acid sequence of a protein specifies the primary structure of a protein. However, the structure has a very large number of conformations of the backbone and side chains. Each polypeptide of the same protein adopts a spontaneously unique structure under conditions of temperature and solubility, which is said to be the native state. The conformation of the native state is determined by the amino acid sequence according to the internal interactions between the backbone, the side chains and the solvent. Also, the pH and ionic composition of the solvent play a factor. However, the function of a given protein does not depend only on the amino acid sequence, but mainly on its overall three dimensional structure

or conformation. It is the fragile balance between the number of forces which eventually determine the protein's conformation and consequently, its functionality. Some of the important key players providing the protein with its unique structure are hydrogen bonds, electrostatic bonds, hydrophobic interactions and van der Waals forces [23].

A hydrogen bond is normally formed between the amide proton and carbonyl oxygen of adjacent peptide groups. These interactions impart the *secondary structure* of proteins. Several hydrogen bonds among the peptide chain may form two basic types of structures: α-helix and β-sheet.

The α-helix structure is a helical arrangement of the peptide planes consisting of 3.6 amino acids forming a single helix turn. Each amino acid residue along the helix axis extends 1.5Å, contributing a total length of 5.4Å along the helix per turn. Each peptide carbonyl is connected by a hydrogen bond to the $N-H$ group four residues further up the chain. All of the H bonds are parallel to the helix axis, the carbonyl groups are pointing in one direction, and the $N-H$ groups are pointing to the opposite direction. Figure 2.5 shows an α-helix structure. The number of residues forming an α-helix structure varies from helix to helix and from protein to protein, but on average it is about 10.

In the β-sheet structure, the peptides form a stripe formation side by side, in which the peptide backbone has a zigzag pattern along the strip. β-sheets can be in either parallel or antiparallel forms. In the parallel β-sheet, adjacent chains run in the same direction. The optimum formation of H bond between parallel residues result in closer distance between residues. Parallel β-sheets are typically large structures, often composed of more than five strands, where hydrophobic side chains appear on both sides of the β-sheet. In contrast, antiparallel β-sheets run in opposite directions, may have only two strands, and their β-sheet strands are usually arranged with all hydrophobic side chains on one side. Figures 2.6 and 2.7 emphasize the difference between these two β-sheets conformations.

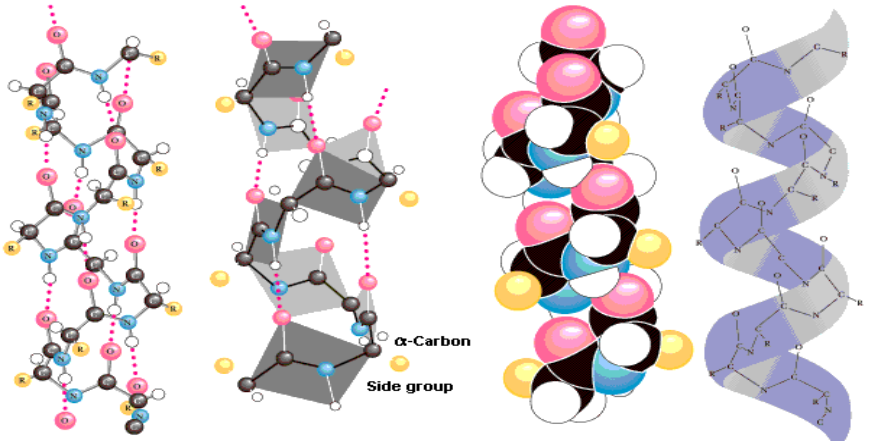


Figure 2.5: α-helix structure representation [11].

There is also a third basic structure type, which is actually not α-helix and not β-sheets which is called *random coils* (or loops).

The three dimensional folding structure of a single polypeptide in a protein is said to be the protein's *tertiary structure*. The primary polypeptide structure already captures all the required information on how to fold the protein into its native structure. The protein chains are normally folded in a way in which the secondary structures are arranged in one of several common patterns, known as motifs. As a rule, proteins normally fold to form the most stable structure. There are protein families which share similar tertiary structure despite their differences in amino acid sequence or functionality.

Many proteins consist of two or more polypeptide chains, referred to as subunits of the protein. The way these subunits are arranged to form one protein and the interactions between them is known as the protein's *quaternary structure* [11]. Figure 2.8 summarizes the four different protein structure types.

Proteins are assigned to one of three global classes based on shape and solubility: fibrous, globular and membrane. Fibrous proteins have a linear structure, often serve structural roles in cells and are insoluble in water. Globular proteins have a spherical shape and are normally involved in metabolic functions. Their polypep-

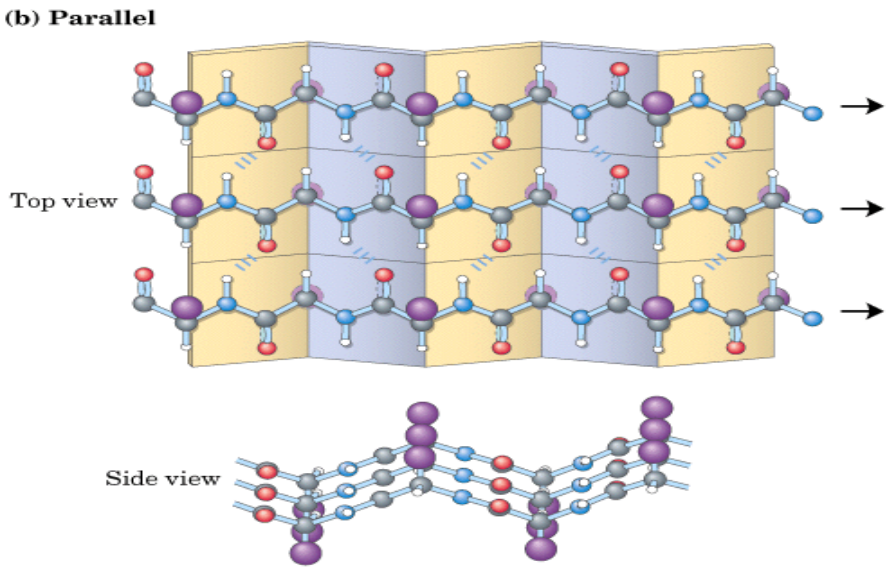


Figure 2.6: Parallel β-sheets structure representation [33].

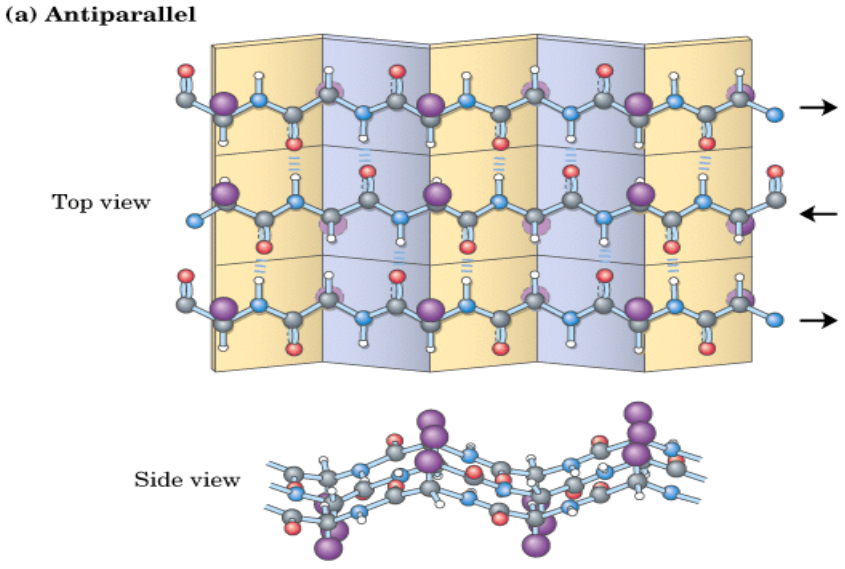


Figure 2.7: Antiparallel β-sheets structure representation [33].

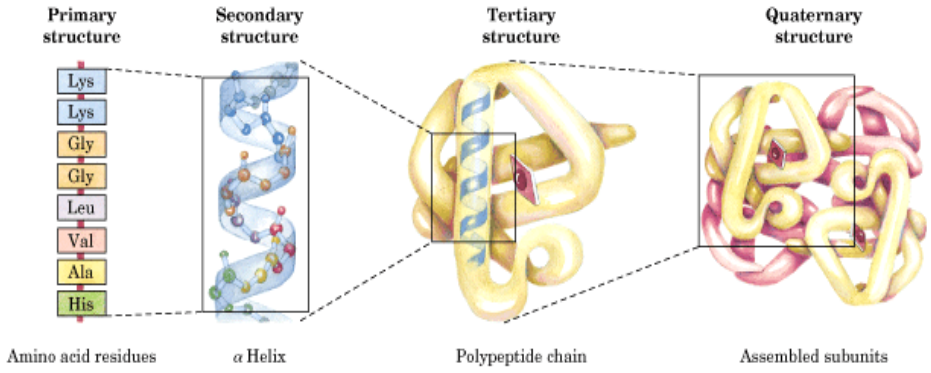


Figure 2.8: Primary, Secondary, Tertiary and Quaternary structures [33].

tide chain is folded in a way so that the hydrophobic amino acid side chains are buried in the interior of the fold and the hydrophilic side chains are on the exterior surface exposed to the solvent. Globular proteins are very soluble in aqueous solutions. Membrane proteins are found within different membrane systems of cells. Membrane proteins have hydrophobic amino acid side chains exposed in their membrane associated regions, and consequently are insoluble in aqueous solutions.

2.2.4 Integral Membrane Proteins (IMPs)

Membranes have an essential role in cellular functionality. They create the boundaries of cells and serve as a surface for many biological reactions. The proteins in the membrane are involved in regulating metabolites, macromolecules and ions. Membranes also take part in the interaction with hormones and many other signaling molecules [11]. In addition, membrane proteins are the most common targets for prescription drugs.

Most membrane proteins are either peripheral or integral proteins. Peripheral proteins are globular proteins which normally interact with the membrane through integral proteins. Integral proteins can be partly embedded or fully embedded within the membrane's monolayers, known as the lipid bilayers. It is estimated

that 20% of human genes encode for integral membrane proteins (IMPs) and some estimates are much higher [48].

Most of IMPs belong to one of two groups. The first group of proteins are attached to the membrane by only small hydrophobic segments, thus are exposed to the solvent on one or both sides of the membrane. The second group consists of proteins with a globular shape which embed in the membrane exposing only a small surface to the solvent outside the membrane. In general, the structure of IMPs within the non polar domain of the membrane are mainly α-helices or β-sheets. In this thesis, our attention is restricted to the major class of membrane proteins, which are bundles of transmembrane α-helices.

Some IMPs have a single transmembrane segment, in which case the segment often appears with a single α-helix structure. Proteins having this form normally function as receptors for extracellular molecules, or as recognition sites for the immune system. In other cases, IMPs may have more than a single transmembrane segment, and form a more globular shape. Proteins from this type may have a number of hydrophobic α-helical segments crossing the membrane back and forth approximately perpendicular to the plane of the lipid bilayer. These proteins often take a role in transport activities across the membrane [11].

2.3 Structure Prediction Using Bioinformatics

Protein structure prediction is an active area of research within bioinformatics. Table 2.2 lists several protein identification applications in bioinformatics.

2.3.1 Identifying Protein Domains

The first task when exploring a protein is to recognize its *domain structure*. Proteins may have a single or several domains. The protein's domain appears as stable, partly independent folding regions within a protein. Some protein domains

Table 2.2: Set of web servers for identifying protein structure in a variety of applications (August, 2005) [38].

Program	URL
Identifying protein domains	
Pfam	http://pfam.wustl.edu/hmmsearch.shtml
Interpro	http://www.ebi.ac.uk/InterProScan/
BLOCKS	http://blocks.fhcrc.org/blocks/blocks_search.html
SMART	http://smart.embl-heidelberg.de/
DomFISH	http://www.bmm.icnet.uk/ 3djigsaw/dom_fish/
Identifying protein secondary structure	
APSSP	http://imtech.res.in/raghava/apssp/
HNN	http://npsa-pbil.ibcp.fr/cgi-bin/npsa_automat.pl?page=npsa_nn.html
Jpred	http://www.compbio.dundee.ac.uk/ www-jpred/
nnPredict	http://www.cmpharm.ucsf.edu/ nomi/nnpredict.html
PredictProtein	http://cubic.bioc.columbia.edu/predictprotein/
PSA	http://bmerc-www.bu.edu/psa/request.htm
PSI-pred	http://insulin.brunel.ac.uk/psiform.html
Phd	http://cubic.bioc.columbia.edu/pp
Prof	http://www.aber.ac.uk/ phiwww/prof/index.html
Identifying protein tertiary structure	
iMoltalk	http://i.moltalk.org/
SDSC1	http://cl.sdsc.edu/hm.html
CPHmodels	http://www.cbs.dtu.dk/services/CPHmodels/
SWISS-MODEL	http://www.expasy.ch/swissmod/SWISS-MODEL.html
Loopp	http://cbsuapps.tc.cornell.edu/
Superfamily	http://supfam.mrc-lmb.cam.ac.uk/SUPERFAMILY/
Wurst	http://www.zbh.uni-hamburg.de/wurst/index.php
Meta Server	http://bioinfo.pl/Meta/
UCLA-DOE FRSVR	http://fold.doe-mbi.ucla.edu/
Identifying sequence motifs and patterns	
Prosite	http://au.expasy.org/tools/scanprosite/
Emotif	http://motif.stanford.edu/emotif/
FingerPRINTScan	http://www.bioinf.man.ac.uk/fingerPRINTScan/
Identifying transmembrane segments	
DAS	http://www.sbc.su.se/ miklos/DAS/
HMMTOP	http://www.enzim.hu/hmmtop/
SOSUI	http://sosui.proteome.bio.tuat.ac.jp/sosuiframe0.html
TMHMM	http://www.cbs.dtu.dk/services/TMHMM/
TMpred	http://www.ch.embnet.org/software/TMPRED_form.html
TopPred	http://bioweb.pasteur.fr/seqanal/interfaces/toppred.html
TopPred2	http://www.sbc.su.se/ erikw/toppred2/

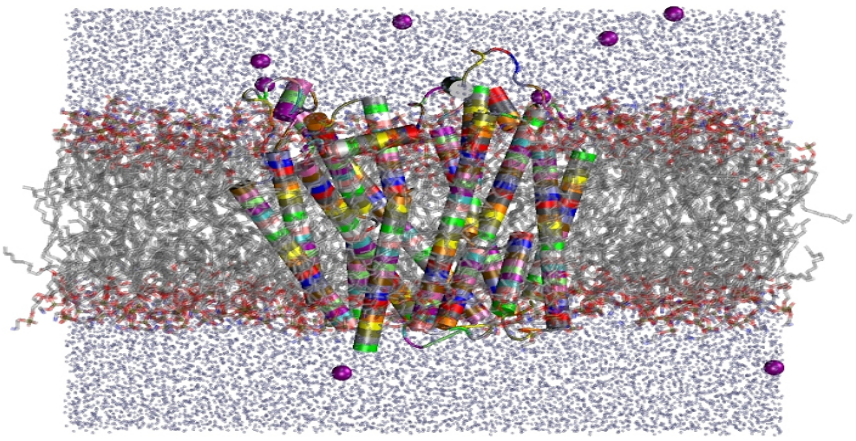


Figure 2.9: Transport of molecules through the cell membrane via membrane proteins [18].

are formed from the same chain of amino acids, while others are formed from discontinuous segments. The implication of knowing the domain boundaries is especially important for exploring the functional and structural characteristics of a protein. In Nuclear Magnetic Resonance (NMR) for example, analyzing a full protein may not be practical in some cases. Similar problems occur in crystallography with linker regions between domains. The advantage of knowing protein domains is also incorporated in structural prediction models. In these cases the prediction performance is increased when predictions are applied on single domain sequences. Most of the available methods for protein domain prediction are based on comparative sequence searches, domain characteristic training and tertiary structure prediction [23].

2.3.2 Secondary Structure Prediction

The main goal of secondary structure prediction is to assign each amino acid with one of three labels: α-helix, β-sheets, and random coils (or loops) corresponding to the structure it appears in. Traditional approaches use preliminary information of different sequence patterns of amino acids. New methods include techniques

like neural networks and hidden Markov models. Today's methods are reaching $75-85\%$ of true secondary structure prediction. Knowing the secondary structure is essential for successful tertiary structure prediction [38].

2.3.3 Tertiary Structure Prediction

The tertiary structure is extremely important when analyzing and predicting a protein's functionality and binding regions from its structure. Structure prediction models normally evolve from one of the main streams: homology modelling, fold recognition and de novo methods. While homology modelling and fold recognition are using existing structures for their prediction, de novo methods are modelling proteins which can not be retrieved from available structure. Homology modelling methods are likely to be used when the sequence has at least 30% of identity. When this is not the case, then folding recognition may be considered.

Both homology modelling and folding recognition search for a homologous template structure from a database of proteins with known structure, and try to align the target sequence with the most similar homologous template sequence. Besides the alignment, tertiary structure predictors are also capable of predicting the target protein with three dimensional model coordinates and assigning a prediction score, describing how well the target sequence is aligned with the homologous templates [38].

2.3.4 Transmembrane Segments Prediction

The task of determining a protein's structure has turned out to be difficult for IMPs. The shortage of data on the structures of membrane proteins from traditional biophysical methods in comparison to water-soluble proteins stems from a lack of success of X-ray crystallography and NMR spectroscopy on these proteins. Structural determination of IMPs can be problematic due to difficulties in obtaining sufficient amounts of sample. A further contributing factor is that

IMPs are generally much more likely to become inactive while handling or waiting for crystallization [22]. The traditional principals of transmembrane segment prediction methods were based on empirical observations that membrane regions are often 20 to 30 residues long, with high hydrophobicity around those regions and are connected with short loops containing positively charged residues. The methods vary in the scoring metrics assigned to the hydrophobicity property, in the prediction algorithm, and rather the prediction is supported by homologous proteins [42]. More recently, a new era of prediction models evolved using more sophisticated mathematical and probability models such as neural networks and hidden Markov models. The most accurate methods claim to predict successfully more than 90% of all membrane regions, with full helix prediction for all proteins higher than 80% [38, 43].

This is an area where data driven methods may be suitable to contribute significantly to the study of structure and function. Therefore, computational methods for IMPs structure predictions are to be highly valued. Transmembrane protein prediction is the major focus of this thesis.

2.4 Summary

Proteins have a central role in all aspects of cell structure and function. All proteins found in nature are composed from a combination of 20 amino acids, and derive their biological function directly from the sequence of amino acid residues in its polypeptide chain. Each polypeptide of the same protein spontaneously adopts a unique structure, which is called the *native state* of the protein. The function of a given protein depends not only on the amino acid sequence, but mainly on its overall three dimensional structure. Several hydrogen bonds among the peptide chain impart the protein's *secondary structure*. The basic types of structure are α-helix, β-sheets, and coils. The three dimensional folding structure of a single polypeptide in a protein is called the protein's *tertiary structure*. Many proteins

consist of two or more polypeptide chains, and the way these chains interact with each other is known as the protein's *quaternary structure*.

Membranes proteins are essential for cellular functionality and are involved in many biological reactions. They are therefore the most common targets for prescription drugs. Integral membrane proteins (IMPs) can be partly embedded or fully embedded within the membrane and the majority of them adopt a α-helix structure.

Different aspects of proteins structural prediction are an active field of study. Determining the structure of IMPs turns out to be problematic using standard methods such as X-ray crystallography and NMR spectroscopy due to the special physical properties of IMPs. Therefore, good transmembrane segment prediction methods are highly valued. While traditional approaches were mainly based on hydrophobicity and polarity scales around membrane regions, new methods are using mathematical and probability models.

This Chapter has introduced the transmembrane segment prediction problem. In the next Chapter we will introduce the necessary techniques from machine learning in order to provide a solution to this problem.

Chapter 3

The Sequential Classification Problem

The sequential classification problem is well known in many different fields such as computational linguistics, speech recognition, and computational biology. All these problems share a common concept: given an observation set of sequences, the goal is to find corresponding label sequences for this observation. The observation sequence can be taken from any domain, e.g: words in a document, different values of stocks over time, nucleotides in a DNA sequence, or amino acids in a protein sequence. The label sequence is the classification of the items in the observation sequence. For instance, in Chapter 2 we introduced proteins and mentioned that proteins consist of a sequence of amino acids. We also said that proteins may have a different secondary structure along their polypeptide, which can be either α-helix, β-sheets or coils. Say we are interested in knowing for each amino acid in the protein sequence what structure it adopts. Then we would like to create a model which assigns a label for each amino acid identifying the protein's secondary structure. Figure 3.1 shows an example of this labelling task.

To perform such a labelling task, different methods have been designed to achieve the same goal: finding the most likely sequence of labels corresponding to the

Figure 3.1: Labelling protein secondary structure.
H = α-Helix, S = β-sheets, C = Coils.

observation based on the given data, known as the training set. The training set conceals the relations between the observation and the label sequences. By incorporating this knowledge into the model, prediction methods are able to come up with a predicted sequence, trying to satisfy the knowledge acquired by the training process. In this thesis we will explore more deeply two main approaches to accomplish this task: the *generative* and the *conditional* models. We will also refer to the difference between *directed* and an *undirected* graphical models and present the main advantages and disadvantages of these approaches.

3.1 Directed Graphical Models

Directed graphical models are a framework for representing graph dependencies between sequences and labels. Formally, we define $G = (V, E)$ to be a directed graph where $v \in V$ corresponds to each of the random variables representing a label Y_v from Y and $e \in E$ corresponds to the transition between a given label to the next one [32]. The direction of G means that every node $v \in V$ has a set of parent nodes, v_Ω, with a topological ordering of the nodes V in G such that nodes in v_Ω appears before v. Therefore, the joint distribution of these random variables is expressed as the product of the local distribution defined over each node and its parents:

$$p(v_1, v_2, ..., v_n) = \prod_{i=1}^{n} p(v_i | v_{\Omega_i})$$

This form of the joint distribution describing the structure of the directed graphical model stems from making the conditional independence assumption: v depends only on the set v_Ω. A case in point is the Hidden Markov Models and the Maximum Entropy Markov Models which we present next.

3.2 Hidden Markov Models (HMMs)

A very common method for solving a sequential classification problem is Hidden Markov Models (HMMs). HMMs are an example of generative models for finding the joint distribution of the paired observation and label sequences. This method uses a probabilistic finite-state automata to find, for a given sequence, the most likely corresponding sequence of labels. HMMs define a joint probability distribution $p(X, Y)$ where X and Y are random variables describing the observation sequence and the labelling sequence respectively. The model is trained to maximize the joint likelihood of the observation and label sequence based on the training sequence. The main drawback in HMMs is that in order to define such a joint distribution, a generative model must calculate all possible observation sequences [45]. Say that we would like to label a secondary structure for a protein sequence with 1000 amino acids. Since there are 20 possible amino acids for each letter in the sequence this task becomes intractable. In order to overcome this problem, HMMs break the input to isolated units, independent of each other. In other words, the observation at any given position in the sequence depends only on the state at that position.

HMMs are represented as a directed graphical model as shown in Figure 3.2. The label distribution of a given state depends on the label distribution of the previous state. In the sequential classification problem that we are trying to solve, each state is associated with a single label $y \in \mathcal{Y}$, where $\mathcal{Y}$ is defined as the label alphabet. A one-to-one relationship between states and labels is not required, and in some models several labels are mapped onto a single state. In the following discussion

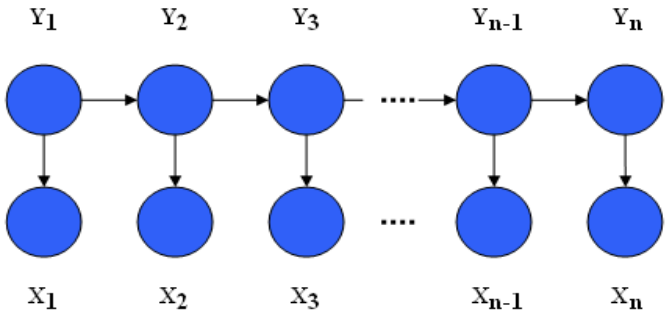


Figure 3.2: Hidden Markov models graphical structure. A blue circle indicates that the variable is generated by the model.

we consider the case of a one-to-one state to label mapping. Say that $y_1, y_2, ..., y_n$ represent the labels at positions $1, 2, ..., n$, then the probability distribution of y_i depends only on the probability distribution of y_{i-1}, and the HMM will estimate the value of $p(y_i|y_{i-1})$ for each $i \in n$ (moving from left to right). Given the current label, HMMs will generate the current observation from that label $p(x_i|y_i)$, and then move to the next label. Similarly, the current observation generated from the current label depends only on that label. Although the observation is known, we are interested in the label which is most likely assigned to that observation. By knowing from the training data the likelihood of assigning possible labels to possible sequences, HMMs are able to generate the observation given the label distribution, and choose the most likely one. Note that in Figure 3.2 all circles are filled, as all graph variables are generated by the model. The joint distribution of HMMs is expressed in the following formula emphasizing the conditional independence relation:

$$p(x, y) = p(y_1)p(x_1|y_1) \prod_{i=2}^{n} p(y_i|y_{i-1})p(x_i|y_i) \tag{3.1}$$

HMMs begin with a start state, and end by reaching the end point of the observation sequence. By looking at the product, we identify two components: $p(y_i|y_{i-1})$

is the transition from the previous label to the current label, and $p(x_i|y_i)$ is the generated observation distribution. This formula relies on the fundamental generative models' independence assumption.

HMM is formally defined by the following:

1. A finite set describing the observation alphabet $\mathcal{X}$.
2. A finite set describing the label alphabet $\mathcal{Y}$.
3. An observation probability distribution $p(x|y)$ representing the probability of generating an observation x given the current state y, where $x \in \mathcal{X}, y \in \mathcal{Y}$.
4. A conditional distribution $p(y|\tilde{y})$ representing the probability of moving from state $\tilde{y}$ to state y, where $y, \tilde{y} \in \mathcal{Y}$.
5. An initial state distribution $p(y)$, where $y \in \mathcal{Y}$.

HMMs solve the sequential classification problem by finding the set of labels which best fits the observation sequence. In other words, finding the label sequence which maximizes the conditional probability of the label given the observation sequence:

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, p(y|x)$$

Since HMMs are defined by the joint distribution of the observation and the label sequences $p(x, y)$, HMMs apply Bayes rule to produce:

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, \frac{p(x, y)}{p(x)}$$

where only $p(x, y)$ is y dependent, and therefore $p(x)$ can be omitted from the maximization problem. The solution of $\hat{y}$ is found using a dynamic programming implementation of the Viterbi algorithm [35].

The Viterbi algorithm is an efficient way of finding the most likely state sequence corresponds to a finite-state discrete-time Markov process. The state sequence estimation problem can also be viewed as the problem of finding the shortest route

through a certain graph. Say our graph contains nodes which represent distinct states at a given time, and arrows which represent transitions from one state to another at the next time unit. The Viterbi algorithm uses recursive steps by which the algorithm determines the shortest path from the initial to the final state. Say c_0 and c_n are the initial and final states respectively, and $C = (c_1, c_2, ..., c_n)$ is the state sequence. Note that every possible state sequence C corresponds to a unique path through the graph. The total length of the path corresponding to some state sequence C is:

$$L(C) = \sum_{k=1}^{n} l(t_k)$$

where $l(t_k)$ is the transition length from c_k to c_{k+1}. The shortest path to node c_k is called the *survivor* corresponding to the node c_k and is denoted as $S(c_k)$. To get to time $(k + 1)$, we need to extend all time-k survivors by one time unit, compute the lengths of the extended path segments, and for each node c_{k+1} find the corresponding survivor for time $(k + 1)$ and store it as $S(c_{k+1})$. The algorithm terminates at time n with the shortest complete path stored as the survivor of $S(c_n)$. This algorithm is a simple version of forward dynamic programming [10].

3.2.1 Sequential Classification Example with HMMs

Now that we have covered the theory behind the HMMs, we would like to apply it on the main task of our thesis, predicting the transmembrane helix regions of a protein. The first step in defining the transmembrane sequential problem in terms of the HMMs model is to specify the following:

1. A finite set describing the observation alphabet $\mathcal{X}$. In our problem, the alphabet of $\mathcal{X}$ is all possible amino acids:
   $(A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y) \in \mathcal{X}$

2. A finite set describing the state alphabet $\mathcal{Y}$. In our case, two states are possible: $(0, 1) \in \mathcal{Y}$, for transmembrane, non-transmembrane helix respectively.

3. An observation probability distribution $p(x|y)$ representing the probability of generating an observation x given the current state y, where $x \in \mathcal{X}, y \in \mathcal{Y}$ (Taken from Table 3.1).

4. A conditional distribution $p(y|\tilde{y})$ representing the probability of moving from state $\tilde{y}$ to state y, where $y, \tilde{y} \in \mathcal{Y}$ (Taken from Table 3.2).

5. An initial state distribution $p(y)$, where $y \in \mathcal{Y}$ (Taken from Table 3.2).

The data we used in this example is taken from a data set consisting of a set of benchmark sequences with experimentally confirmed transmembrane regions compiled by Möller et al. [30].[1] We only included proteins with a high level of trust (assigned with transmembrane annotation trust level of A to C, as was suggested by Möller et al.) and which are significantly different, based on pairwise similarity clustering. The resulting set consists of 148 transmembrane protein sequences. After analyzing the data, we calculated the conditional probability distributions as displayed in Tables 3.1 and 3.2.

We now demonstrate how HMMs solve the labelling prediction task on a small example. The same solution is applicable in a real sequence labelling task. Say we would like to predict the labelling sequence of a "micro" protein having the sequence: $MFINR$. Maximizing the joint distribution of HMMs expressed in Equation (3.1), can be calculated using a dynamic programming technique. To solve this maximization problem, the sequence $MFINR$ is broken up into 5 iterations. Each iteration solves the local maximum problem of assigning a label to the observation at location i. Tables 3.3 and 3.4 describe the dynamic programming calculation process.

After solving the local maximization problem of each iteration and moving to the next one, when we solve the last iteration we actually get the solution on the whole sequence. In this example, the label that maximizes the joint distribution of HMMs on the sequence $MFINR$ is: 00000 as shown in Table 3.4.

[1] The data set can be accessed via ftp://ftp.ebi.ac.uk/databases/testsets/transmembrane

Table 3.1: Conditional probability distribution of amino acids and transmembrane helix labels calculated from a data set compiled by Möller et al.

Amino Acid (X)	Label=0	Label=1	Total	$P(X\|Y=0)$	$P(X\|Y=1)$	$P(Y=0\|X)$	$P(Y=1\|X)$
A	3239	1753	4992	0.0813	0.1157	0.6488	0.3512
C	431	213	644	0.0108	0.0141	0.6693	0.3307
D	2045	129	2174	0.0513	0.0085	0.9407	0.0593
E	2509	129	2638	0.0630	0.0085	0.9511	0.0489
F	1514	1260	2774	0.0380	0.0832	0.5458	0.4542
G	2905	1356	4261	0.0729	0.0895	0.6818	0.3182
H	814	141	955	0.0204	0.0093	0.8524	0.1476
I	2054	1678	3732	0.0516	0.1108	0.5504	0.4496
K	2303	114	2417	0.0578	0.0075	0.9528	0.0472
L	3647	2542	6189	0.0916	0.1678	0.5893	0.4107
M	1007	621	1628	0.0253	0.0410	0.6186	0.3814
N	1686	248	1934	0.0423	0.0163	0.8718	0.1282
P	2101	409	2510	0.0528	0.0270	0.8371	0.1629
Q	1818	184	2002	0.0456	0.0121	0.9081	0.0919
R	2301	151	2452	0.0578	0.0100	0.9384	0.0616
S	2740	830	3570	0.0688	0.0548	0.7685	0.2325
T	2318	780	3098	0.0582	0.0515	0.7482	0.2518
V	2575	1694	4269	0.0647	0.1118	0.6032	0.3978
W	597	392	989	0.0150	0.0259	0.6036	0.3964
Y	1224	523	1747	0.0307	0.0345	0.7006	0.2994
Total	39828	15147	54975	1	1		

Table 3.2: Conditional probability of transition between helical states calculated from a data set compiled by Möller et al.

	$p(y_i = 0\|y_{i-1} = 0) = 0.7114$	$p(y_i = 1\|y_{i-1} = 0) = 0.0126$
	$p(y_i = 0\|y_{i-1} = 1) = 0.0131$	$p(y_i = 1\|y_{i-1} = 1) = 0.2629$
Total	$p(y_i = 0) = 0.7245$	$p(y_i = 1) = 0.2755$

Table 3.3: Maximizing HMMs joint distribution using dynamic programming.

Iteration	Maximization problem
1	$max\Big(p(y_1 = 0)p(x_1 = M\|y_1 = 0), p(y_1 = 1)p(x_1 = M\|y_1 = 1\Big)$
2	$max\Big(p(y_2 = 0\|y_1)p(x_2 = F\|y_2 = 0), p(y_2 = 1\|y_1)p(x_2 = F\|y_2 = 1\Big)$
3	$max\Big(p(y_3 = 0\|y_2)p(x_3 = I\|y_3 = 0), p(y_3 = 1\|y_2)p(x_3 = I\|y_3 = 1\Big)$
4	$max\Big(p(y_4 = 0\|y_3)p(x_4 = N\|y_4 = 0), p(y_4 = 1\|y_3)p(x_4 = N\|y_4 = 1\Big)$
5	$max\Big(p(y_5 = 0\|y_4)p(x_5 = R\|y_5 = 0), p(y_5 = 1\|y_4)p(x_5 = R\|y_5 = 1\Big)$

Table 3.4: Solving the maximization problem using dynamic programming.

Iteration	Maximization problem	Y Value
1	$max\Big(p(y_1 = 0)p(x_1 = M\|y_1 = 0), p(y_1 = 1)p(x_1 = M\|y_1 = 1\Big) = max(0.0183, 0.0113)$	$y_1 = 0$
2	$max\Big(p(y_2 = 0\|y_1 = 0)p(x_2 = F\|y_2 = 0), p(y_2 = 1\|y_1 = 0)p(x_2 = F\|y_2 = 0\Big) = max(0.0270, 0.0005)$	$y_2 = 0$
3	$max\Big(p(y_3 = 0\|y_2 = 0)p(x_3 = I\|y_3 = 0), p(y_3 = 1\|y_2 = 0)p(x_3 = I\|y_3 = 0\Big) = max(0.0367, 0.0007)$	$y_3 = 0$
4	$max\Big(p(y_4 = 0\|y_3 = 0)p(x_4 = N\|y_4 = 0), p(y_4 = 1\|y_3 = 0)p(x_4 = N\|y_4 = 0\Big) = max(0.0301, 0.0005)$	$y_4 = 0$
5	$max\Big(p(y_5 = 0\|y_4 = 0)p(x_5 = R\|y_5 = 0), p(y_5 = 1\|y_4 = 0)p(x_5 = R\|y_5 = 0\Big) = max(0.0411, 0.0007)$	$y_5 = 0$

From the formal definition of HMMs, it is clear that HMMs rely on the fundamental generative models independent assumption. The probability of entering a current state depends only on the probability of leaving the previous state $p(y_i|y_{i-1})$, and the generated observation distribution $p(x_i|y_i)$ depends only on the current state. This assumption may be inappropriate in some sequential problem cases where the observation sequence has long-range dependencies between observations. By breaking the sequence to several isolated units, we might lose important data which is described by these dependencies, and essential to describe the problem accurately. Therefore, it is appropriate to use different techniques which have the ability to relax these strong independence assumptions on the observation data.

3.3 Generative and Conditional Models

The difficulties due to the problem which described in the previous paragraph led to an alternative approach using conditional models instead. While generative models define a joint probability distribution of the observation and the labelling

sequences $p(X, Y)$, the conditional models specify the probability of a label given an observation sequence $p(Y|X)$. Thus, no effort is spent on modelling all possible observation sequences, but only on selecting the labels which maximize the conditional probability $p(Y|X)$, while relaxing strong independence assumptions on the observation data [21]. Furthermore, while conditional models are trained to minimize the labelling errors, generative models are trained to maximize the joint probability of the training data. Finding the joint probability might be useful if the trained model is also used to generate the data, but might not be so accurate if the actual data was not generated by the model, as is often the case in practice [37].

Conditional models can use several, not necessarily independent, overlapping features which describe the constraints on the problem [37]. These features describe different constraints on the model, taken from the real-world. For example, if we return to the protein's structure problem, performance is improved dramatically by adding sets of features which take to account polarity and hydrophobicity attributes of each amino acid. Maximum Entropy Markov Models (MEMMs) are an example of conditional models.

3.4 Maximum Entropy Markov Models (MEMMs)

MEMMs are a conditional model approach for the sequential classification problem. While HMMs and other generative models calculate the joint probability distribution of the observation and label sequences by generating the observation, MEMMs finds the label distribution conditioned by the observation. In other words, instead of having two probability distributions: one for moving from a previous state to current state $p(y_i|y_{i-1})$, and one for generating the observation conditioned on the current state $p(x_i|y_i)$, MEMMs have a single probability distribution in the form $p(y_i|x_i, y_{i-1})$. It is to say that the label assigned at time t depends only on the observation at time t and the previous label at time $t - 1$. This prob-

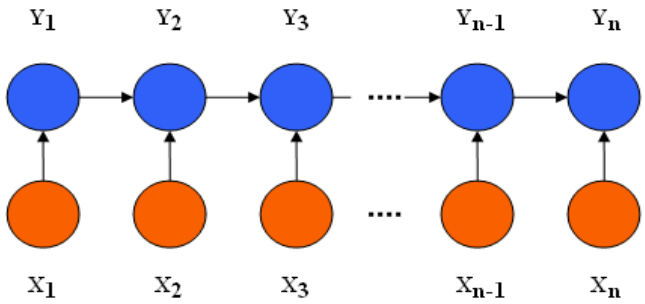


Figure 3.3: Maximum entropy Markov models graphical structure. An orange circle indicates that the variable is not generated by the model.

ability is represented in the MEMMs graph structure over label sequence y given the observation sequence x and is formulated as:

$$p(y|x) = p(y_1|x_1) \prod_{i=2}^{n} p(y_i|y_{i-1}, x_i) \tag{3.2}$$

Equation (3.2) can be displayed as a graph as shown in Figure 3.3.

The key principal of MEMMs is by selecting a particular distribution, when no other knowledge to constrain the choice of parameters is available. Entropy of a probability distribution is the measure of uncertainty. The entropy is maximized when the distribution is as close to uniform distribution as possible. If we are interested in measuring the probability distribution of a finite training set and imply it on a test set, then according to the maximum entropy principal: The probability distribution that mostly reflects the training set with the ability to generalize to unseen data, is one which has the maximum entropy subject to the set of constraints that were pre-defined in the problem . In other words, the main principle behind the MEMMs approach is to create the most uniform model that satisfies all known constraints and makes the least assumptions about unseed data [8].

The fact that the observations are now events to be conditioned upon rather than generated involve the creation of non-dependent overlapping features on the ob-

servation sequence. The conditional probability of the label given the observation can be now defined as a log linear transition function:

$$p(y|x) = \frac{1}{Z(x,y)} exp(\sum_j \lambda_j F_j(y,x))$$

where F_j is a feature function, $Z(x,y)$ is a normalization factor, and λ_j are the estimated parameters describing the feature's weight [29]. Every possible label is defined as a state and has an exponential distribution. Given the observations, the model is trained based on pre-defined features to predict the most likely distribution for the next state. Feature functions will be discussed further in Section 4.3. The parameter estimation may be implemented using different algorithms such as Generalized Iterative Scaling and Improved Iterative Scaling [21].

Similar to HMMs, MEMMs solve the sequential classification problem by finding the set of labels which best fit the observation sequence. In other words, finding the label sequence which maximize the conditional probability of the label given the observation sequence:

$$\hat{y} = \underset{y}{\text{argmax}}\, p(y|x)$$

The solution of $\hat{y}$ is found using a dynamic programming implementation of the Viterbi algorithm [35].

3.5 MEMMs Label Bias Problem

The main weakness of MEMMs and other non-generative finite-state models based on directed graphical models is called the *label bias problem* [21]. In this problem, the probability of leaving a given state competes against the probability of entering any potential next state instead of competing against the most likely sequence of all consecutive states together. Say that every transition from one state to another has a score, defined by the conditional probability of possibly entering the next state given the current state and the observation sequence. Since the total transition score of all incoming transitions is equal to the total score of all outgoing

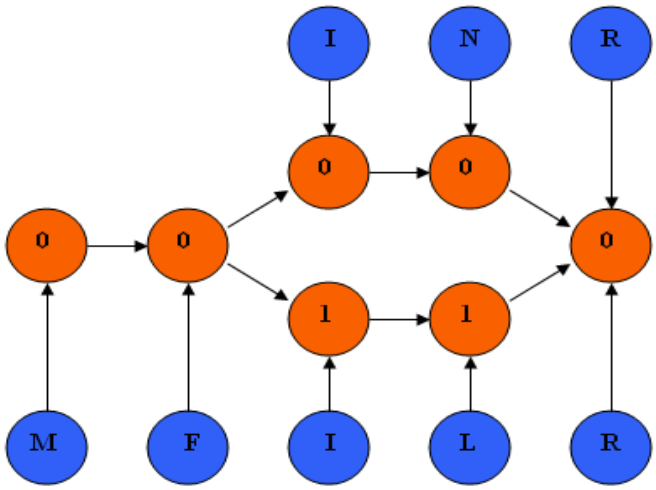


Figure 3.4: Label bias problem example. A blue circle indicates the observation sequence x, and an orange circle indicates the label assigned by the model y.

transitions, an observation may affect what percentage of the total incoming score might be directed to each one of possible outgoing transitions. Nevertheless, the total outgoing score must be preserved among all possible outgoing transitions. Hence, it may cause a bias towards states with fewer outgoing transitions, which benefit from a relatively high conditional probability of being in those states since their total outgoing score is shared among fewer next states. The Markovian assumptions in MEMMs and similar state-conditional models separate the decision making at one step from future dependent decisions of consecutive steps, and therefore suffer from the label bias problem. The following example will make things clearer. Say our MEMM is trained with the following two sequences: First sequence is $MFINR$ with label sequence of 00000. Second sequence is $MFILR$ with label sequence of 00110. Figure 3.4 describes the graph structure of MEMMs after the training is completed. Suppose we are interested in determining the most likely label sequence given a new observation sequence, we use Equation (3.2)

and maximize the conditional distribution of $p(y|x)$. Now lets look at Figure 3.4 more carefully. The model was trained with two sequences who start with the observation MF, assigned with the labels 00. Therefore, the joint distribution of the only possible label sequence 00 given the observation sequence MF is:

$$p(y_1 = 0, y_2 = 0|x_1 = M, x_2 = F) = p(y_1 = 0|x_1 = M)p(y_2 = 0|y_1 = 0, x_2 = F)$$

The next observation I matches both sequences and therefore, the mass accumulated so far is distributed among the two possible labels 0 and 1. If we look further on the path $MFIN$ with next observation N, we can see that according to the graph structure it is assigned with label 0. In this case, the number of outgoing transitions is exactly 1. Therefore, we conclude that

$$p(y_4 = 0|y_3 = 0, x_4 = N) = p(y_4 = 0|y_3 = 0) = 1$$

In other words, the probability of assigning the label sequence $y_1y_2y_3y_4 = 0000$ given the previous label sequence $y_1y_2y_3 = 000$ is 1! In this case, the observation sequence in x_4 is ignored completely by the model. Similarly, this happens with observation L and label 1 on the $MFIL$ path. In both cases, the decision of entering the next state is already made by leaving the current state, regardless of the observation sequence. The explanation of this phenomenon is due to the fact that the number of outgoing transitions is one. It is noteworthy that it does not necessarily have to be the case in order to suffer from the label bias problem. Say there are a number of outgoing transitions, and one of them is more frequent than the others. The probability of choosing this transition path is always greater than the other transitions, and this may result in ignoring the observation sequence along this path. A more general way where the label bias problem may occur is where the probability of all observations X is $p(y_i|y_{i-1}, X) \approx p(y_i|y_{i-1})$.

Conditional Random Fields (CRFs) are a probabilistic framework for labelling sequential data which overcome the label bias problem. Similarly to MEMMs, CRFs are belonging to the same family of conditional models. CRFs are a form of undirected graphical state models that define a log-linear distribution for each

state over the label sequence based on the observation sequence [45]. The major difference between MEMMs and CRFs stems from the directed vs. non-directed approach. MEMMs find for each state the exponential models for the conditional probabilities of next states given the current state, while CRFs have a single exponential model for the joint probability of the entire sequence of labels given the observation sequence [21].
In Chapter 5 we conduct a set of experiments comparing the prediction results between MEMMs vs. CRFs approaches showing the outperforming of CRFs. In the next Chapter, we cover the theory behind the CRFs model.

3.6 Summary

The sequential classification problem is defined as finding corresponding label sequences given an observation set of sequences. Different prediction models try to fulfil this task by training the model on a set of known observation and labelling sequences, known as the training set. In the training process, the prediction model captures this knowledge from the training set, and comes up with a suggested prediction sequence, satisfying the constraints acquired in the training. There are two main approaches to accomplish this task: the *generative* and the *conditional* models.

Hidden Markov Models (HMMs) are an example of generative models based on finding the joint distribution of the paired observation and label sequences $p(X, Y)$. The main drawback in HMMs is that in order to define such a joint distribution, a generative model should calculate all possible observation sequences, which becomes impractical in complex problems. To overcome this problem, HMMs break the input to isolated units, independent from one another. This assumption may be inappropriate in some sequential problems where long-range dependencies exist between observations.

Conditional models are an alternative approach which do not calculate the joint

distibution. Instead of finding the joint probability distribution $p(X, Y)$, conditional models specify the probability of a label given an observation sequence $p(Y|X)$. Instead of modelling all possible observation sequences, conditional models select the labels which maximize the conditional probability $p(Y|X)$. Maximum Entropy Markov Models (MEMMs) are an example of conditional models. Entropy of a probability distribution is the measure of uncertainty. The key principal of MEMMs is based on the entropy assumption: creating a model that satisfies all known constraints but treats the unknown uniformly. MEMMs have a single probability distribution in the form $p(y_i|x_i, y_{i-1})$, meaning that the label assigned at time t depends only on the observation and the previous label at time $t-1$. Experiments have shown that MEMMs suffer from a weakness known as the *label bias problem* [21]. The Markovian assumptions in MEMMs separate the decision making at one step from future dependent decisions of consecutive steps, and therefore may cause a bias towards states with fewer outgoing transitions.

Conditional Random Fields (CRFs) are an approach which overcomes the disadvantages of both HMMs and MEMMs. CRFs are the main topic of the next Chapter.

Chapter 4

Conditional Random Fields (CRFs)

Conditional Random Fields (CRFs) are a probabilistic framework from the family of conditional models for labelling sequential data. CRFs are a form of undirected graphical state models that define a single log-linear distribution for the entire label sequence conditioned on the observation sequence [45]. The main advantage of CRFs over Maximum Entropy Markov Models (MEMMs) and other conditional finite-state directed models, is that they overcome a weakness called the *label bias problem*. This problem defines the condition when a model is biased towards low entropy transition states. Since MEMMs use a per-state exponential model conditioned on the previous state, MEMMs may suffer from the label bias problem, as explained in more details in Section 3.5. CRFs also allow for a seamless and modular integration of biological domain knowledge expressed as model constraints.

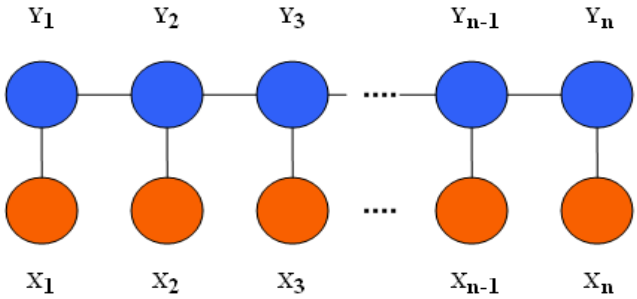


Figure 4.1: CRFs graphical structure. A blue circle indicates that the variable is generated by the model.

4.1 Undirected Graphical Models

CRFs are an undirected graphical model, conditioned on the random variable representing the observation sequence X. Formally, we define $G = (V, E)$ to be an undirected graph to represent the labels in the observation sequence. Here $v \in V$ is a set of nodes corresponds to random variables representing the labels and observations, and $e \in E$ is a set of undirected edges between nodes corresponds to the transition between a given label to the next one. Note that according to the definition of conditional independence for undirected graphical models, if two vertices v_i and v_j do not share edge between them in graph G, then the random variables v_i and v_j are conditionally independent given all other random variables in the graph. Even though in theory the structure of a graph G may be arbitrary, in our application the graph is a simple chain where each node corresponds to a label [45]. Figure 4.1 demonstrates the undirected graph structure of CRFs. The edges between the label nodes Y correspond to the cliques between these random variables. The observation nodes X do not share edges as the model does not assume any independence assumptions among the observation.

4.2 CRF Definition and Model Derivation

Let $G = (V, E)$ be a graph, $Y = (Y_v)$ $v \in V$ be a family of finite-valued random variables, and $X = (X_v)$ $v \in V$ be a family of functions.

Definition 1 *(Y, X) is a conditional random field (CRF) if*

1. $P(Y|X) > 0$ $\forall Y$
2. *For all $v \in V$*

$$P(Y_v|Y_{V-v}, X) = P(Y_v|Y_{N(v)}, X)$$

where $N(v)$ are the neighbors of v in G. This is called the Markov property.

Definition 2 *Y is a Gibbs field if*

$$P(Y) = \frac{1}{Z} \exp(-\frac{1}{T} U(Y))$$

where Z is a normalizing constant, T is a constant called the temperature (which we assume is equal to one), and U is called the energy.

The energy U can be expressed as:

$$U(Y) = \sum_{c \in C} U_c(Y)$$

where C is the set of all cliques (completely connected subgraphs) in G, and U_c is the energy for a particular clique.

According to the Hammersley-Clifford Theorem [24], a CRF is equivalent to a Gibbs field, thus

$$P(Y|X) \propto \exp\{U(Y, X)\} = \exp\{\sum_{c \in C} U_c(Y, X)\}^1 \tag{4.1}$$

For sequential data, the graph G is a chain and therefore the clique set C consists of vertices (C_1) and edges (C_2), thus

$$U(Y, X) = \sum_{\{v\} \in C_1} V_1(Y_v, X) + \sum_{\{v,w\} \in C_2} V_2(Y_v, Y_w, X)$$

[1] The minus sign has been absorbed in the function.

Using the notation of Lafferty [21] we can re-write Equation (4.1) as:

$$p_\theta(y|x) \propto exp\left(\sum_{e\in E,j} \lambda_j f_j(e, y|_e, x) + \sum_{v\in V,j} \mu_j g_j(v, y|_v, x)\right)$$

where x is a data sequence, y a label sequence, $y|_S$ is the set of components of y associated with the vertices in subgraph S. The vectors f and g represent the local features with corresponding weight vectors λ and μ.
The joint distribution can be expressed in a slightly different form:

$$p_\theta(y|x) \propto exp\left(\sum_j \lambda_j f_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k g_k(y_i, x, i)\right) \tag{4.2}$$

$f_j(y_{i-1}, y_i, x, i)$ is a transition feature function of the entire observation sequence and the labels at positions i and $i-1$ in the label sequence. $g_k(y_i, x, i)$ is a state feature function of the entire observation sequence and the label at position i in the label sequence. λ_j and μ_k are estimated from the training data. We assume that the feature functions f_k and g_k are given and fixed [44].

4.3 Feature Functions and Model Estimation

The joint distribution of the labels $Y_v \in Y$ may be factorized into a normalized product of real-valued potential functions that are conditionally independent. Each potential function corresponds to a feature whose support is a clique of the graph. When the graph is a chain, as in our case, the cliques are the vertices and the edges of the graph [34].

The feature functions constrain the conditional probability distribution $p(y|x)$. The satisfaction of a constraint increases the likelihood of the global configuration. Note that no feature independence assumption is made, and several dependent overlapping features are allowed. Different weights assigned to the parameters associated with the features, can be used to distinguish between important and rather features.

An example of a feature is:

$$u(x,i) = \begin{cases} 1 & \text{if the amino acid at position } i \text{ is polar} \\ 0 & \text{otherwise} \end{cases}$$

If the current state in a state function, or the current and previous state in a transition function are satisfied then the feature function takes on the value $u(x,i)$ [44]. For example, the feature function $f_j(y_{i-1}, y_i, x, i)$ is assigned with the return value of the function $u(x,i)$, in case that the label at position $i-1$ corresponds to a protein secondary structure of α-helix, and the label at position i to a secondary structure of β-strand:

$$f_j(y_{i-1}, y_i, x, i) = \begin{cases} u(x,i) & \text{if } y_{i-1} = \alpha \text{ helix and} \\ & y_i = \beta \text{ strand} \\ 0 & \text{otherwise} \end{cases}$$

In what follows, we generalize the transition functions to include state functions by writing:

$$g(y_i, x, i) = f(y_{i-1}, y_i, x, i)$$

We also define the sum of a feature over the sequence by:

$$f_j(x,y) = \sum_{i=1}^{n} f_j(y_{i-1}, y_i, x, i)$$

where $f_j(y_{i-1}, y_i, x, i)$ refers to either transition or state function [44]. Therefore, the probability of a label sequence y given the observation sequence x is of the form:

$$p_\lambda(y|x) = \frac{1}{Z_\lambda(x)} exp(\sum_j \lambda_j f_j(x,y)) \tag{4.3}$$

where

$$Z_\lambda(x) = \sum_y exp(\sum_j \lambda_j f_j(x,y)) \tag{4.4}$$

The most likely label sequence $\hat{y}$ for a given input sequence x is:

$$\hat{y} = \underset{y}{\text{argmax}}\, p(y|x,\lambda) = \underset{y}{\text{argmax}} \sum_j \lambda_j \cdot f_j(x,y)$$

The most probable $\hat{y}$ can be found efficiently using the Viterbi algorithm [37].

The parameters of the CRFs model are calculated using the principal of the maximum entropy theory. The entropy of a probability distribution is the measure of uncertainty. Entropy is maximized when the distribution is close to uniform as possible. In other words, the main principle is specify a model that satisfies all known constraints but treats the unknown constraints uniformly [17]. The conditional entropy of $p(y|x)$ is:

$$H(Y|X) = -\sum_{x,y} \tilde{p}(x,y) log p(y|x)$$

where $\tilde{p}(x,y)$ is the empirical joint distribution of the observation and the label. Using the maximum entropy principal one can show that the expectation of the features in the data is equal to the expectation of the features in the model [21]:

$$\sum_{x,y} \tilde{p}(x) p(y|x) F(x,y) = E_{\tilde{p}(x,y)} F(x,y)^{\dagger}$$

The maximum entropy distribution that satisfies the feature constraints from the empirical distribution is revealed using an estimation technique. In the next Section we will use this equation to find the parameters of the model using maximum log-likelihood.

4.4 The Maximum Likelihood Approach

The parameter estimation problem is to determine the parameters $\theta = (\lambda_1, \lambda_2, ...; \mu_1, \mu_2, ...)$ in Equation 4.2 from the training data $T = (x^{(k)}, y^{(k)})_{k=1}^{N_{prot}}$ with empirical distribution $\widetilde{p}(x,y)$ [21]. In what follows, we rewrite θ as λ as we are using $f_j(x,y)$ to represent both the transition and the state function. Assuming that the training data T is independently and identically distributed, then the product of the probabilities over as a function of λ, is known as the *likelihood*.

[†]Here $F(x,y)$ represents all the features and the parameters specified in the model.

In maximum likelihood training, the goal is to find parameters which maximize the likelihood. However, since the logarithm is an increasing function, and the log of the products is the sum of the logarithms, it is convenient to maximize the log-likelihood. Several methods are available to find the parameters which maximize the log-likelihood. Some of the methods such as iterative scaling algorithm [2] and its variants were designed specifically to solve the particular form of the maximum entropy likelihood. However, a consensus is emerging that a more general form of optimization techniques like the Quasi-Newton Methods are more efficient [27, 36].

The log-likelihood of a CRF model $p_\lambda(y|x)$ given a joint empirical distribution $\tilde{p}(x, y)$ is defined as:

$$\mathcal{L}_{\tilde{p}}(\lambda) = \sum_{x,y} \tilde{p}(x, y) log p_\lambda(y|x) \tag{4.5}$$

where

1. $\mathcal{L}_{\tilde{p}}(\lambda) \leq 0$
   Note that $p_\lambda(y|x) > 0 \; \forall y$ according to the CRFs definition.

2. $\mathcal{L}_{\tilde{p}}(\lambda) = 0$ is the optimal value of $\mathcal{L}_{\tilde{p}}(\lambda)$ if and only if $p_\lambda(y|x) = 1$ and $\tilde{p}(x, y) > 0$.

The maximum likelihood problem is defined as:

$$\hat{\lambda} = \underset{\lambda}{\text{argmax}} \, \mathcal{L}_{\tilde{p}}(\lambda)$$

$\mathcal{L}_{\tilde{p}}(\lambda)$ is a concave function of λ which guarantees the existence of a unique global maximum [2].

By substituting $p_\lambda(y|x)$ as defined in Equation (4.3) into Equation (4.5) we get the CRFs general log-likelihood objective function:

$$\mathcal{L}_{\tilde{p}}(\lambda) = \sum_{x,y} \tilde{p}(x, y) \sum_j \lambda_j f_j(x, y) - \sum_x \tilde{p}(x) log \sum_y exp \sum_j \lambda_j f_j(x, y) \tag{4.6}$$

Differentiating the log-likelihood by λ_i gives:

$$\frac{\partial \mathcal{L}_{\tilde{p}}(\lambda)}{\partial \lambda_j} = \sum_{x,y} \tilde{p}(x,y) f_j(x,y) - \sum_{x,y} \tilde{p}(x) p_\lambda(y|x) f_j(x,y)$$

$$= E_{\tilde{p}(x,y)} f_j(x,y) - E_{\tilde{p}(x) p_\lambda(y|x)} f_j(x,y)$$

$E_{\tilde{p}(x,y)} f_j(x,y)$ denotes the expectation with respect to empirical distribution $\tilde{p}$. $E_{\tilde{p}(x) p_\lambda(y|x)} f_j(x,y)$ denotes the expectation with respect to the model distribution $\tilde{p}(x) p_\lambda(y|x)$ [3].

Since the maximum-likelihood function is concave over the space of λ, it has a global maximum when the derivative is zero. Setting this derivative to zero yields the maximum entropy model constraint: the expectation of feature $f_j(x,y)$ with respect to the model distribution $\tilde{p}(x) p_\lambda(y|x)$ is equal to the expected value of feature $f_j(x,y)$ under the empirical distribution obtained from the training data $\tilde{p}(x,y)$ [37].

4.5 Parameter Estimation

To reiterate, the main goal is to determine the parameters $\hat{\lambda} = (\lambda_1, \lambda_2, ...\lambda_n) \in \mathbb{R}^n$ which maximize the function 4.6. In practice, this task is computationally expensive, as the number of features may be very large, for example, in our case nearly one million. As we have stated before, there are several methods available to determine $\hat{\lambda}$, including Generalized Iterative Scaling [8], Improved Iterative Scaling [2], Newton's Method, and Quasi-Newton's Methods such as Limited Memory BFGS [25]. In the following sections we describe, in some details, about each of these techniques. The commonality among all these techniques is that they are iteratively trying to find a new set of parameters: $\lambda + \Delta = (\lambda_1 + \delta_1, \lambda_2 + \delta_2, ...\lambda_n + \delta_n)$ which increase the log-likelihood function as much as possible. After a new set is found, the parameters are updated: $\lambda \leftarrow \lambda + \Delta$ and the algorithm continues to the next iteration. This process is terminated either when convergence is reached (the

update value of Δ is smaller than a threshold), or a maximum pre-defined number of iterations is exceeded. While all parameter estimation algorithms share the same principal of trying to improve the target log-likelihood function at each iteration, they differ in the method of calculating the improvement, noted as Δ [27].

4.5.1 Iterative Scaling

The main goal is to find $\lambda + \Delta = (\lambda_1 + \delta_1, \lambda_2 + \delta_2, ...\lambda_n + \delta_n)$, which increases the log-likelihood target function as much as possible. We can express this change in the log-likelihood function by using Equation (4.5) as:

$$\mathcal{L}_{\tilde{p}}(\lambda + \Delta) - \mathcal{L}_{\tilde{p}}(\lambda) = \sum_{x,y} \tilde{p}(x,y) log p_{\lambda+\Delta}(y|x) - \sum_{x,y} \tilde{p}(x,y) log_{\lambda}(y|x)$$

By substituting $log p_{\lambda+\Delta}(y|x)$ as defined in Equation (4.3) we get:

$$= \sum_{x,y} \tilde{p}(x,y) \sum_{j} \delta_j f_j(x,y) - \sum_{x} \tilde{p}(x) log \frac{Z_{\lambda+\Delta}(x)}{Z_{\lambda}(x)}$$

We can lower bound $\mathcal{L}_{\tilde{p}}(\lambda + \Delta) - \mathcal{L}_{\tilde{p}}(\lambda)$ by using the observation

$-log\ \alpha \geq 1 - \alpha$ (true for all $\alpha > 0$) with α equal to $log \frac{Z_{\lambda+\Delta}(x)}{Z_{\lambda}(x)}$ and get:

$$\mathcal{L}_{\tilde{p}}(\lambda + \Delta) - \mathcal{L}_{\tilde{p}}(\lambda) \geq \sum_{x,y} \tilde{p}(x,y) \sum_{j} \delta_j f_j(x,y) + 1 - \sum_{x} \tilde{p}(x) \frac{Z_{\lambda+\Delta}(x)}{Z_{\lambda}(x)}$$

$$= \sum_{x,y} \tilde{p}(x,y) \sum_{j} \delta_j f_j(x,y) + 1 - \sum_{x} \tilde{p}(x) \frac{\sum_y exp \sum_j (\lambda_j + \delta_j) f_j(x,y)}{\sum_y exp \sum_j \lambda_j f_j(x,y)}$$

$$= \sum_{x,y} \tilde{p}(x,y) \sum_{j} \delta_j f_j(x,y) + 1 - \sum_{x} \tilde{p}(x) \sum_{y} p_{\lambda}(y|x) exp(\sum_{j} \delta_j f_j(x,y))$$

$$= \mathcal{A}(\Delta|\lambda) \tag{4.7}$$

Back to our main goal, we have just shown that finding Δ such that as $\mathcal{A}(\Delta|\lambda) > 0$ will improve the log-likelihood target function, as $\mathcal{L}_{\tilde{p}}(\lambda + \Delta) - \mathcal{L}_{\tilde{p}}(\lambda) \geq \mathcal{A}(\Delta|\lambda)$.

Seemingly, we could maximize the desired Δ by differentiating $\mathcal{A}(\Delta|\lambda)$ with respect to each δ_j. Then update each parameter by $\lambda_j \leftarrow \lambda_j + \delta_j$ and increase the log-likelihood on every iteration, as long as the solution to $\mathcal{A}(\Delta|\lambda) > 0$ can be found. However, if we look more carefully at Equation (4.7), we realize that δ_j appears inside an exponential expression $exp \sum_j \delta_j f_j(x, y)$ and by differentiating Equation (4.7) with respect to δ_k we get:

$$\frac{\partial \mathcal{A}(\Delta)}{\partial \delta_k} = \sum_{x,y} \tilde{p}(x, y) f_k(x, y) - \sum_x \tilde{p}(x) \sum_y p_\lambda(y|x) f_k(x, y) exp \sum_j \delta_j f_j(x, y) \quad (4.8)$$

and the exponential form of the second term, depends on the sum expression of δ_j. This prevents us from factoring out δ_j from the above equation. This problem has prompted the development of two iterative scaling approaches, which overcome this problem by refining the expression $\mathcal{A}(\Delta|\lambda)$ in Equation (4.7). These are the Generalized Iterative Scaling (GIS) [8] and the Improved Iterative Scaling (IIS) [2] methods.

4.5.2 Generalized Iterative Scaling (GIS) Algorithm

This parameter estimation technique is based on improving the value of δ_j's by a factor proportional to the ratio of the expectation with respect to empirical distribution $E_{\tilde{p}(x,y)} f_j(x, y)$ and the expectation with respect to the distribution $E_{\tilde{p}(x) p_\lambda(y|x)} f_j(x, y)$. GIS relies on the maximum entropy approach in which the model reaches its convergence when the empirical expectation is equal to the model distribution:

$$E_{\tilde{p}(x,y)} f_j(x, y) = E_{\tilde{p}(x) p_\lambda(y|x)} f_j(x, y)$$

GIS defines a constant C as the maximal number of active features for any possible sequences x, y (not only in the training data).

$$C = \max_{x,y} \sum_j f_j(x, y)$$

In addition, the GIS algorithm adds a new constraint to the model, such that for each possible sequence (x, y), the sum of all features sums to C

$$\forall x, y \sum_j f_j(x, y) = C$$

This constraint, $\check{f}(x, y)$, can be defined as:

$$\check{f}(x, y) = C - f_j(x, y)$$

The correction feature is added to the existing set of features, and is activated in case that the sum of all activated features for that particular sequence x, y is lower than C [28].

Now we can express Equation (4.7) together with constant C as:

$$\mathcal{A}(\Delta|\lambda) = \sum_{x,y} \tilde{p}(x, y) \sum_j \delta_j f_j(x, y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\lambda(y|x) exp(C \sum_j \frac{\delta_j f_j(x, y)}{C}) \tag{4.9}$$

Note that $\frac{f_j(x,y)}{C}$ is a probability distribution function, since the denominator C is defined as the sum of all features, and the numerator is the feature $f_j(x, y)$ which are both always non-negative. By summing the numerator over j, x, y we get the sum of all features $f_j(x, y)$ divided by C which adds to one. Therefore, we can use Jensen's inequality for probability distribution functions (p.d.f):

$$exp \sum_x p(x)q(x) \leq \sum_x p(x) exp(q(x))$$

and by referring to $(\frac{f_j(x,y)}{C})$ as $p(x)$ and $(\delta_j C)$ as $q(x)$ we can lower bound $\mathcal{A}(\Delta|\lambda)$ in Equation (4.9) as:

$$\mathcal{A}(\Delta|\lambda) \geq \sum_{x,y} \tilde{p}(x, y) \sum_j \delta_j f_j(x, y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\lambda(y|x) \sum_j (\frac{f_j(x, y)}{C}) exp(\delta_j C)$$
$$= \mathcal{B}(\Delta|\lambda)$$

Now differentiating with respect to δ_k:

$$\frac{\partial \mathcal{B}(\Delta)}{\partial \delta_k} = \sum_{x,y} \tilde{p}(x,y) f_k(x,y) - \sum_x \tilde{p}(x) \sum_y p_\lambda(y|x) f_k(x,y) exp(\delta_k C) \tag{4.10}$$

Finally, by taking the log on both sides we find the update rule of GIS:

$$\delta_j = log(\frac{E_{\tilde{p}(x,y)} f_j(x,y)}{E_{\tilde{p}(x) p_\lambda(y|x)} f_j(x,y)})^{\frac{1}{C}}$$

It can be easily noticed that the convergence speed and the improvement factor δ_j of each parameter λ_j depends on C. The higher C is, the slower the step to convergence. In many cases, the sum of all enabled features for a given row in the training data is not equal to a constant C, and a correction feature is required. Adding such a correction feature slows the convergence. Therefore, an alternative approach was developed, which calculates the step without a correction feature.

4.5.3 Improved Iterative Scaling (IIS) Algorithm

The Improved Iterative Scaling solution for this problem is by defining F as the sum of all features for the given sequences x, y:

$$F(x,y) = \sum_j f_j(x,y)$$

If we are dealing with binary features only, then F can be specified as as the sum of all activated features over sequences x, y. By incorporating F in Equation (4.7) we get:

$\mathcal{A}(\Delta|\lambda) =$

$$\sum_{x,y} \tilde{p}(x,y) \sum_j \delta_j f_j(x,y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\lambda(y|x) exp(F(x,y) \sum_j \frac{\delta_j f_j(x,y)}{F(x,y)}) \tag{4.11}$$

Note that $\frac{f_j(x,y)}{F(x,y)}$ is a probability distribution function (By summing the numerator over j we get one). Therefore, we can use Jensen's inequality for p.d.f:

$$exp(\sum_x p(x) q(x)) \leq \sum_x p(x) exp(q(x))$$

and by substituting $\frac{f_j(x,y)}{F(x,y)}$ as $p(x)$ and $\delta_j F(x,y)$ as $q(x)$ we can express Equation (4.11) as:
$\mathcal{A}(\Delta|\lambda) \geq$

$$\sum_{x,y} \tilde{p}(x,y) \sum_j \delta_j f_j(x,y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\lambda(y|x) \sum_j (\frac{f_j(x,y)}{F(x,y)}) exp(\delta_j F(x,y))$$

$$= \mathcal{B}(\Delta|\lambda) \tag{4.12}$$

So by using the lower bound $\mathcal{B}(\Delta|\lambda)$ we can improve the log-likelihood function as:

$$\mathcal{L}_{\tilde{p}}(\lambda + \Delta) - \mathcal{L}_{\tilde{p}}(\lambda) \geq \mathcal{B}(\Delta|\lambda)$$

$\mathcal{B}(\Delta|\lambda)$, however, can be differentiated over δ_k [3]:

$$\frac{\partial \mathcal{B}(\Delta)}{\partial \delta_k} = \sum_{x,y} \tilde{p}(x,y) f_k(x,y) - \sum_x \tilde{p}(x) \sum_y p_\lambda(y|x) f_k(x,y) exp(\delta_k F(x,y)) \tag{4.13}$$

This time we can calculate the desired Δ by differentiating over $\mathcal{B}(\Delta|\lambda)$ with respect to each δ_k. In contrast to Equation (4.7), in Equation (4.13) δ_k appears alone inside the exponential expression $exp\delta_k F(x,y)$ and does not depend on other values of δ. Therefore, δ_k can be estimated easily in a small number of iterations using the Newtown-Raphson method [27].
First, we solve $\frac{\partial \mathcal{B}(\Delta)}{\partial \delta_k}$ for each parameter λ_k. Then we update $\lambda_k \leftarrow \lambda_k + \delta_k$ and increase the log-likelihood on every iteration, as long as the solution to $\mathcal{B}(\Delta|\lambda) > 0$ can be found until convergence is reached.

4.5.4 Newton's Method

Newton's method is a classical approach for solving optimization problems. Suppose we want to minimize the function $f : \mathbb{R}^\kappa \rightarrow \mathbb{R}$

$$\min_{x \in \mathbb{R}^n} f(x)$$

which is twice continuously differentiable. In order to do so, we first solve

$$\nabla f(x) = 0 \tag{4.14}$$

where $\nabla f(x)$ is the gradient of f. Then we verify that the Hessian at the solution point is positive definite. The Hessian is an $n \times n$ matrix defined as:

$$H = \Big(\frac{\partial^2 f}{\partial x_i x_j}\Big)_{i,j=1\ldots n}$$

In many cases, it may not be possible to find an analytical solution of Equation 4.14. This is the case where numerical methods are used. Given the solution at $\bar{x}$, Newton's methods uses Taylor's expansion to approximate the value of the function in the neighborhood of $\bar{x}$:

$$\nabla f(x) = \nabla f(\bar{x}) + \nabla^2 f(\bar{x})(x - \bar{x}) + o(\|x - \bar{x}\|) \tag{4.15}$$

By omitting $o(\|x - \bar{x}\|)$, we define $l(x)$ to be:

$$l(x) = \nabla f(\bar{x}) + \nabla^2 f(\bar{x})(x - \bar{x}) \tag{4.16}$$

The principal of Newton's method is to solve the equation $l(x) = 0$ instead of Equation (4.14), and using the solution as the next step in the iteration via the formula:

$$x_{i+1} = x_i - \nabla^2 f(x_i)^{-1} \nabla f(x_i) \tag{4.17}$$

A geometric explanation for this step is shown in Figure 4.2 for the one dimensional function $p(x)$. Let $l(X)$ be the gradient of this function. From Equation 4.16, the gradient $l(X)$ to $p(x)$ at x_1 is:

$$l(x_1) = p(x_1) + \nabla p(x_1)x - \nabla p(x_1)x_1$$

$l(x)$ meets the x-axis at x_2, and therefore:

$$p(x_1) + \nabla p(x_1)x_2 - \nabla p(x_1)x_1 = 0$$

$$x_2 = x_1 - \frac{p(x_1)}{\nabla p(x_1)}$$

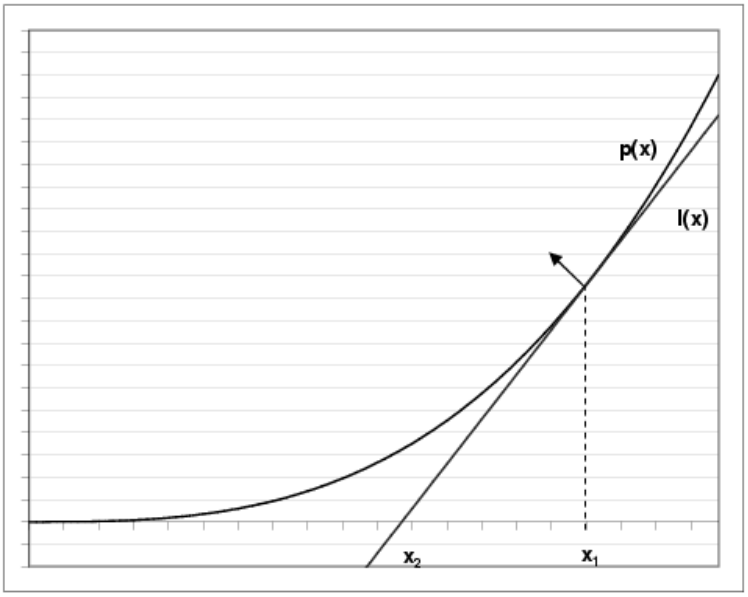


Figure 4.2: Newton's method for approximating the solution of equation $p(x) = 0$. See text for explanation.

Similarly, since for us $p(x) = \nabla f(x)$ and $\nabla p(x) = \nabla^2 f(x)$, the general form is:

$$x_{i+1} = x_i - \nabla^2 f(x_i)^{-1} \nabla f(x_i)$$

as defined in Equation (4.17).
The difference d_i in the x value in every iteration is defined as:

$$d_i = -\nabla^2 f(x_i)^{-1} \nabla f(x_i)$$

The Newton's algorithm is as follows:

1. Choose x_0 and set $i = 0$
2. Calculate $\nabla f(x_i)$. If $\nabla f(x_i) = 0$ then stop
3. Calculate $\nabla^2 f(x_i)$. If $\nabla^2 f(x_i)$ is not positive definite then Goto 1
4. Calculate $d_i = -\nabla^2 f(x_i)^{-1} \nabla f(x_i)$
5. Update $x_{i+1} = x_i + d_i$ and $i++$

6. Goto 2

Updating x_{i+1} in every iteration does not guarantee a function improvement $f(x_{i+1}) < f(x_i)$, but if $\nabla^2 f(x_i)$ is positive definite, then the change in the value of x_i is considered to be in a descent direction, named the Newton's search direction. Therefore, the line search method can be integrated into the Newton's algorithm to ensure that $f(x_{i+1}) < f(x_i)$ in every step:

1. Choose x_0 and set $i = 0$
2. Calculate $\nabla f(x_i)$. If $\nabla f(x_i) = 0$ then stop
3. Calculate $\nabla^2 f(x_i)$. If $\nabla^2 f(x_i)$ is not positive definite then stop
4. Calculate $d_i = -\nabla^2 f(x_i)^{-1} \nabla f(x_i)$
5. Find α_i which $f(x_i + \alpha_i d_i) = \min_\alpha f(x_i + \alpha d_i)$
6. Update $x_{i+1} = x_i + \alpha_i d_i$ and $i++$
7. Goto 2

Step 3 requires that the Hessian $\nabla^2 f(x_i)$ is positive definite. A way to overcome this requirement is by defining a positive constant γ_i so $\nabla^2 f(x_i) + \gamma_i I$ is positive definite, and I is the identity matrix. Then we can replace step 4 with:

$$d_i = -(\nabla^2 f(x_i) + \gamma_i)^{-1} \nabla f(x_i)$$

[31].

4.5.5 Quasi-Newton Methods

A careful examination at Newton's algorithm reveals that the Hessian of $f(x_i)$ has to be inverted on every iteration, a task which is computationally expensive if the dimension of the problem is high. Quasi-Newton methods are avoiding

the computation of the Hessian matrix by estimating the Hessian using numerical methods.

After the first iteration, both $\nabla f(x_0)$, $\nabla^2 f(x_0)$ and x_1 are calculated. From substituting x_0, x_1 into Equation (4.15) we get:

$$\nabla f(x_0) - \nabla f(x_1) = \nabla^2 f(x_1)(x_0 - x_1) + o(\|x_0 - x_1\|)$$

Thus,

$$\nabla^2 f(x_1)^{-1}(\nabla f(x_0) - \nabla f(x_1)) = (x_0 - x_1)$$

The Quasi-Newton condition avoids the calculation of $\nabla^2 f(x_1)^{-1}$ by replacing it with H_1. In general form, Quasi-Newton condition is written as:

$$H_{i+1}\gamma_i = \delta_i \tag{4.18}$$

where
$H_{i+1} = \nabla^2 f(x_{i+1})^{-1}$
$\gamma_i = \nabla f(x_{i+1}) - \nabla f(x_i)$
$\delta_i = x_{i+1} - x_i$

After finding such a matrix H_{i+1}, the search direction d_{i+1} is defined as:

$$d_{i+1} = -H_{i+1}\nabla f(x_{i+1}) \tag{4.19}$$

There are a number of variations of Quasi-Newton methods, each of which define a different way of calculating H_{i+1} [31]. We next briefly describe the BFGS methods as this method outperformed other parameter estimation methods on Maximum Entropy models [27].

4.5.6 Limited Memory BFGS Method

We have used the LBFGS [25] parameter estimation technique in our transmembrane prediction model with CRFs. The code relies on sparse matrix operations

available from the COLT distribution [14] and an implementation of the Quasi-Newton optimization algorithm (LBFGS) available under the Riso open source project [9].

According to the Quasi-Newton BFGS method, the update of H_{i+1} is in the form:

$$H_{i+1} = H_i + \Big(1 + \frac{\gamma_i^T H_i \gamma_i}{\delta_i^T \gamma_i}\Big)\frac{\delta_i \delta_i^T}{\delta_i^T \gamma_i} - \Big(\frac{\delta_i \gamma_I^T H_i + H_i \gamma_i \delta_i^T}{\delta_i^T \gamma_i}\Big) \tag{4.20}$$

After the Hessian is estimated, the direction can be calculated easily using Equation (4.19) [31].

4.6 Conditional Probability in Matrix Form

When the underlying graph is a chain, as in our case, we can express the conditional probability over a label sequence Y, given the observation sequence X, in a matrix form. We define two new states, start and stop as: $Y_0 = start$ and $Y_{n+1} = stop$. For each position i in the observation sequence x , define the $|\mathcal{Y}|\times|\mathcal{Y}|$ matrix random variable $M_i(x) = M_i(y_{i-1}, y_i|x)$ by

$$M_i(y_{i-1}, y_i|x) = exp\Big(\sum_j \lambda_j f_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k g_k(y_i, x, i)\Big)$$

where $y_{i-1}, y_i \in \mathcal{Y}$, and $\mathcal{Y}$ is the label alphabet. The above can be simplified by referring to transition functions as a general case of state functions and expressed as:

$$M_i(y_{i-1}, y_i|x) = exp\Big(\sum_j \lambda_j f_j(y_{i-1}, y_i, x, i)\Big)$$

Since conditional models do not try to model all possible observation sequences x ,in contrast to generative models, these matrices can be computed directly for a given observation sequence x and the parameter vector λ. The normalization function $Z(x)$ is defined as the [start,stop] *entry* calculated on the product:

$Z(x) = \left[M_1(x)M_2(x)...M_{n+1}(x)\right]_{start,stop}$. Using this notation, the conditional probability of a label sequence y is written as

$$p(y|x, \lambda) = \frac{\prod_{i=1}^{n+1} M_i(y_{i-1}, y_i|x)}{Z(x)}$$

where $y_0 = start$ and $y_{n+1} = stop$ [21]. This conditional probability can be maximized using a dynamic programming implementation of the Viterbi algorithm [35].

4.7 Feature Integration with the Model

The most important aspect of specifying the model is selecting a set of features that capture the relationships between the observation and the label sequences, in our case the protein sequence and the helical structure respectively [4]. The number of different features which can be applied to a model is infinite, and therefore, it is a hard task to find the right feature combination which describes the problem as in reality. Assembling such a feature combination which satisfies the problem is an empirical process which involves many experiments. On each experiment, a combination of features is selected and then the model is evaluated based on its prediction. It is possible to evaluate and score the prediction based on the selected feature combination.

In this Section we present the different features which we have used in our experiments. Some of the features we have already introduced in our previous work [26], and the others are being introduced for the first time in this thesis. We have extended the selected set of features to capture biological constraints and divided them into 8 different types of groups:

1. Start, Edge, End features
2. Basic Amino Acid features
3. Single Side Neighboring features
4. Single Side Shuffled Neighboring features
5. Double Side Neighboring features
6. Double Side Shuffled Neighboring features
7. Amino Acids Property features
8. Border features

4.7.1 Start, End and Edge Features

By using these features we capture the probability of starting/ending a sequence with a given label or the transition probability for moving from one state to the consecutive state. For instance, the start unigram feature has the form:

$$u_{start}(x,i) = \begin{cases} 1 & \text{if the Amino Acid in sequence } x \\ & \text{at position } i \text{ is the first in the sequence} \\ 0 & \text{otherwise} \end{cases}$$

The relationship between the observation and the two possible structures, helix membrane/non-helix membrane, is described in the feature:

$$f_{start_H}(y_i,x,i) = \begin{cases} u_{start}(x,i) & \text{if } y_i = \text{Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{start_{NH}}(y_i,x,i) = \begin{cases} u_{start}(x,i) & \text{if } y_i = \text{Non-Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

The Edge feature in contrast, is a bigram feature which depends on two consecutive labels:

$$f_{edge_{H-H}}(y_{i-1},y_i,x,i) = \begin{cases} u_{edge}(x,i) & \text{if } y_{i-1} = \text{Helix membrane} \\ & \text{and } y_i = \text{Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{edge_{H-NH}}(y_{i-1},y_i,x,i) = \begin{cases} u_{edge}(x,i) & \text{if } y_{i-1} = \text{Helix membrane} \\ & \text{and } y_i = \text{Non-Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{edge_{NH-H}}(y_{i-1},y_i,x,i) = \begin{cases} u_{edge}(x,i) & \text{if } y_{i-1} = \text{Non-Helix membrane} \\ & \text{and } y_i = \text{Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{edge_{NH-NH}}(y_{i-1}, y_i, x, i) = \begin{cases} u_{edge}(x,i) & \text{if } y_{i-1} = \text{Non-Helix membrane} \\ & \text{and } y_i = \text{Non-Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

4.7.2 Basic Amino Acid Features

Amino acids have different tendencies to appear in a membrane helical structure. Since our language contains 20 possible amino acids, we have 20 different unigram features from this type. The unigram feature of amino acid n in position i is:

$$u_n(x,i) = \begin{cases} 1 & \text{if the Amino Acid in sequence } x \\ & \text{at position } i \text{ is from type } n \\ 0 & \text{otherwise} \end{cases}$$

Using this unigram, a feature for describing the relationship between the observation and the two possible structures has the form:

$$f_{n_H}(y_i, x, i) = \begin{cases} u_n(x,i) & \text{if } y_i = \text{Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{n_{NH}}(y_i, x, i) = \begin{cases} u_n(x,i) & \text{if } y_i = \text{Non-Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

4.7.3 Single Side Neighboring Amino Acid Features

While the basic amino acid feature captures the different tendencies of a given amino acid to appear in a membrane helical structure, we are also interested in capturing the tendency of the same amino acid given its adjacent neighbors. Say our current amino acid is A_i, we have created a feature to capture the tendency of A_i to appear in a helix given A_{i-1}. In the same way we have also created additional features of A_i given A_{i-1} ,A_{i-2}, up to k-feature of A_i given A_{i-1}, ..., A_{i-k}. The same is applicable for the opposite single side of A_i given A_{i+1}, ..., A_{i+k}.

4.7.4 Single Side Shuffled Neighboring Amino Acid Features

Similar to Single Side Neighboring Amino Acid Features, but this time we are interested in capturing the tendency of the same amino acid given its adjacent neighbors without being concerned about their order.

Say our current amino acid is A_i, we have created a feature to capture the tendency of A_i to appear in a helix given A_{i-1}. In the same way we have also created additional features of A_i given $A_{i-1} \cup A_{i-2}$, up to k-feature of A_i given $A_{i-1}\cup$, ..., $\cup A_{i-k}$. The same is applicable for the opposite single side of A_i given $A_{i+1}\cup$, ..., $\cup A_{i+k}$.

The motivation of creating the shuffled features is based on the hypothesis that the location of the transmembrane regions are determined by the difference in the amino acid distribution in various structural parts of the protein rather than by specific amino acid composition of these parts. We test this hypothesis in Section 5.3.2 by comparing the use of Single Side Neighboring Amino Acid Features vs. Single Side Shuffled Neighboring Amino Acid Features on a set of benchmark membrane helix sequences.

4.7.5 Double Side Neighboring Amino Acid Features

We have also captured the tendency of an amino acid to appear given its adjacent neighbors from both sides together. Say our current amino acid is A_i, we define a feature which captures the tendency of A_i to appear in a helix given A_{i-1},...,A_{i-k} and A_{i+1},...,A_{i+k}.

The multigram feature of amino acid n in position i is:

$$m_{i-1,i,i+1}(x,i) = \begin{cases} 1 & \text{if the Amino Acid in sequence } x \text{ at} \\ & \text{position } (i-1), i, (i+1) \text{ is from} \\ & \text{type } m_{(i-1)}, m_{(i)}, m_{(i+1)} \\ & \text{respectively} \\ 0 & \text{otherwise} \end{cases}$$

Using this multigram, a feature for describing the relationship between the observation and the two possible structures has the form:

$$f_{m_H}(y_i, x, i) = \begin{cases} m_{i-1,i,i+1}(x,i) & \text{if } y_i = \text{Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{m_{NH}}(y_i, x, i) = \begin{cases} m_{i-1,i,i+1}(x,i) & \text{if } y_i = \text{Non-Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

4.7.6 Double Side Shuffled Neighboring Amino Acid Features

Similar to Double Side Neighboring Amino Acid Features, but this time we are interested in capturing the tendency of an amino acid to appear given it's adjacent neighbors from both sides together without being concerned about their order in which they appear.

Say our current amino acid is A_i, we define a feature which captures the tendency of A_i to appear in a helix given $A_{i-1} \cup ,...,\cup A_{i-k}$ and $A_{i+1} \cup ,...,\cup A_{i+k}$.

The multigram feature of amino acid n in position i is:

$$m_{i-1,i,i+1}(x,i) = \begin{cases} 1 & \text{if the Amino Acid in sequence } x \text{ at} \\ & \text{position } (i-1), i, (i+1) \text{ is from} \\ & \text{type } m_{(i-1)} \cup m_{(i)} \cup m_{(i+1)} \\ & \text{respectively} \\ 0 & \text{otherwise} \end{cases}$$

Using this multigram, a feature for describing the relationship between the observation and the two possible structures has the form:

$$f_{m_H}(y_i, x, i) = \begin{cases} m_{i-1,i,i+1}(x, i) & \text{if } y_i = \text{Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{m_{NH}}(y_i, x, i) = \begin{cases} m_{i-1,i,i+1}(x, i) & \text{if } y_i = \text{Non-Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

4.7.7 Amino Acid Property Features

Amino acids differ from one another in their chemical structure expressed by their side chains. The fact that amino acids from the same classification group appear in similar locations, motivated us to create special property features. We have adopted the classification from Sternberg [39], classifying the amino acids into nine groups[2], each group described by a unigram feature. Note that some amino acids may appear in more than one group simultaneously.

In the following Section, we explain in more details how we selected the feature combination for our model.

The hydrophobicity property for instance, is described in the feature:

$$u_{Hydrophobic}(x, i) = \begin{cases} 1 & \text{if the Amino Acid in sequence } x \text{ at} \\ & \text{position i} \in \text{(M,I,L,V,A,G,F,W,Y,H,K,C)} \\ 0 & \text{otherwise} \end{cases}$$

The relationship between observation and label for hydrophobicity property is given by:

$$f_{hydrophobic_H}(y_i, x, i) = \begin{cases} u_{Hydrophobic}(x, i) & \text{if } y_i = \text{Helix} \\ & \text{membrane} \\ 0 & \text{otherwise} \end{cases}$$

[2]Aromatic (F,W,Y,H), Hydrophobic (M,I,L,V,A,G,F,W,Y,H,K,C), Positive (H,K,R), Polar (W,Y,C,H,K,R,E,D,S,Q,N,T), Charged (H,K,R,E,D), Negative (E,D), Aliphatic (I,L,V), Small (V,A,G,C,P,S,D,T,N), Tiny (A,G,S)

$$f_{hydrophobic_{NH}}(y_i, x, i) = \begin{cases} u_{Hydrophobic}(x, i) & \text{if } y_i = \text{Non-Helix} \\ & \text{membrane} \\ 0 & \text{otherwise} \end{cases}$$

4.7.8 Border Features

By using these features we capture the border between a sequence of amino acids labelled with one structure and a sequence labelled with another. The relationship between the observation sequence labelled with the two possible structures, helix membrane/non-helix membrane, is described in the feature:

$$f_{border_{H-NH}}(y_i, x, i) = \begin{cases} 1 & \text{if } y_{i-j}, ..., y_{i-1} = \text{Helix membrane} \\ & \text{and } y_i, ..., y_{i+j} = \text{Non-Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{border_{NH-H}}(y_i, x, i) = \begin{cases} 1 & \text{if } y_{i-j}, ..., y_{i-1} = \text{Non-Helix membrane} \\ & \text{and } y_i, ..., y_{i+j} = \text{Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

Table 4.1: The list of all features used in the prediction model.

Feature Name	Feature Condition†	Label Condition‡
Start feature H	AA at position i is the first in the sequence	y_i = H
Start feature NH	AA at position i is the first in the sequence	y_i = NH
Edge feature H:H	AA at position i is neither first nor last in the sequence	y_{i-1} = H, y_i = H
Edge feature H:NH	AA at position i is neither first nor last in the sequence	y_{i-1} = H, y_i = NH
Edge feature NH:H	AA at position i is neither first nor last in the sequence	y_{i-1} = NH, y_i = H
Edge feature NH:NH	AA at position i is neither first nor last in the sequence	y_{i-1} = NH, y_i = NH
End feature H	AA at position i is the last in the sequence	y_i = H
End feature NH	AA at position i is the last in the sequence	y_i = NH
Basic Amino Acid feature H	AA at position i is from type n	y_i = H
Basic Amino Acid feature NH	AA at position i is from type n	y_i = NH
Single Side Neighboring Left H	AA at position $i, ..., (i-j)$ is from type $m_{(i)}, ..., m_{(i-j)}$, $j = 1...k$	y_i = H
Single Side Neighboring Left NH	AA at position $i, ..., (i-j)$ is from type $m_{(i)}, ..., m_{(i-j)}$, $j = 1...k$	y_i = NH
Single Side Neighboring Right H	AA at position $i, ..., (i+j)$ is from type $m_{(i)}, ..., m_{(i+j)}$, $j = 1...k$	y_i = H
Single Side Neighboring Right NH	AA at position $i, ..., (i+j)$ is from type $m_{(i)}, ..., m_{(i+j)}$, $j = 1...k$	y_i = NH
Single Side Shuffled Neighboring Left H	AA at position $i, ..., (i-j)$ is from type $m_{(i)}\cup, ..., \cup m_{(i-j)}$, $j = 1...k$	y_i = H
Single Side Shuffled Neighboring Left NH	AA at position $i, ..., (i-j)$ is from type $m_{(i)}\cup, ..., \cup m_{(i-j)}$, $j = 1...k$	y_i = NH
Single Side Shuffled Neighboring Right H	AA at position $i, ..., (i+j)$ is from type $m_{(i)}\cup, ..., \cup m_{(i+j)}$, $j = 1...k$	y_i = H
Single Side Shuffled Neighboring Right NH	AA at position $i, ..., (i+j)$ is from type $m_{(i)}\cup, ..., \cup m_{(i+j)}$, $j = 1...k$	y_i = NH
Double Side Neighboring H	AA at position $(i-j), ..., (i+j)$ is from type $m_{(i-j)}, ..., m_{(i+j)}$, $j = 1...k$	y_i = H
Double Side Neighboring NH	AA at position $(i-j), ..., (i+j)$ is from type $m_{(i-j)}, ..., m_{(i+j)}$, $j = 1...k$	y_i = NH
Double Side Shuffled Neighboring H	AA at position $(i-j), ..., (i+j)$ is from type $m_{(i-j)}\cup, ..., \cup m_{(i+j)}$, $j = 1...k$	y_i = H
Double Side Shuffled Neighboring NH	AA at position $(i-j), ..., (i+j)$ is from type $m_{(i-j)}\cup, ..., \cup m_{(i+j)}$, $j = 1...k$	y_i = NH
Hydrophobic feature H	AA at position i is from type (M,I,L,V,A,G,F,W,Y,H,K,C)	y_i = H
Hydrophobic feature NH	AA at position i is from type (M,I,L,V,A,G,F,W,Y,H,K,C)	y_i = NH
Aromatic feature H	AA at position i is from type (F,W,Y,H)	y_i = H
Aromatic feature NH	AA at position i is from type (F,W,Y,H)	y_i = NH
Positive feature H	AA at position i is from type (H,K,R)	y_i = H
Positive feature NH	AA at position i is from type (H,K,R)	y_i = NH
Polar feature H	AA at position i is from type (W,Y,C,H,K,R,E,D,S,Q,N,T)	y_i = H
Polar feature NH	AA at position i is from type (W,Y,C,H,K,R,E,D,S,Q,N,T)	y_i = NH
Charged feature H	AA at position i is from type (H,K,R,E,D)	y_i = H
Charged feature NH	AA at position i is from type (H,K,R,E,D)	y_i = NH
Negative feature H	AA at position i is from type (E,D)	y_i = H
Negative feature NH	AA at position i is from type (E,D)	y_i = NH
Aliphatic feature H	AA at position i is from type (I,L,V)	y_i = H
Aliphatic feature NH	AA at position i is from type (I,L,V)	y_i = NH
Small feature H	AA at position i is from type (V,A,G,C,P,S,D,T,N)	y_i = H
Small feature NH	AA at position i is from type (V,A,G,C,P,S,D,T,N)	y_i = NH
Tiny feature H	AA at position i is from type (A,G,S)	y_i = H
Tiny feature NH	AA at position i is from type (A,G,S)	y_i = NH
Border Non Helix-Helix	AA at position $(i-j), ..., (i+j)$ is from type $m_{(i-j)}, ..., m_{(i+j)}$, $j = 1...k$	$y_{i-j}, ..., y_{i-1}$ = NH, $y_i, ..., y_{i+j}$ = H
Border Helix-Non Helix	AA at position $(i-j), ..., (i+j)$ is from type $m_{(i-j)}, ..., m_{(i+j)}$, $j = 1...k$	$y_{i-j}, ..., y_{i-1}$ = H, $y_i, ..., y_{i+j}$ = NH

AA = Amino-Acid, *H* = Helix, *NH* = Non-Helix.

† Feature is enabled only if *feature condition* holds

‡ If *label condition* holds then the feature value is 1, 0 otherwise.

4.8 CRFs Prediction Example

We now give a simple example of transmembrane helix prediction. For this example we will assume, for simplicity, that all proteins are 4 amino acids long, and the amino acids alphabet is composed of the amino acids $(A, C, D, E, F) \in \mathcal{X}$. The labelling alphabet describes if the current amino acid is inside a helix membrane/non-helix membrane region and denoted by $(0, 1) \in \mathcal{Y}$.

We define two property groups:

$$(A, C, F) \in Hydrophobic$$

$$(C, D, E) \in Polar$$

Note: the amino acid C is both Hydrophobic and Polar.

We create the following unigram features:

$$u_{Basic}(x, i) = \begin{cases} 1 & \text{if the Amino Acid in sequence } x \\ & \text{at position } i \text{ is from type } n \\ 0 & \text{otherwise} \end{cases}$$

$$u_{Hydrophobic}(x, i) = \begin{cases} 1 & \text{if the Amino Acid at position } i \in \text{(A,C,F)} \\ 0 & \text{otherwise} \end{cases}$$

$$u_{Polar}(x, i) = \begin{cases} 1 & \text{if the Amino Acid at position } i \in \text{(C,D,E)} \\ 0 & \text{otherwise} \end{cases}$$

Using these unigrams, each feature for describing the relationship between the observation and the two possible structures has the form:

$$f_{n_H}(y_i, x, i) = \begin{cases} u_n(x, i) & \text{if } y_i = \text{Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

$$f_{n_{NH}}(y_i, x, i) = \begin{cases} u_n(x, i) & \text{if } y_i = \text{Non-Helix membrane} \\ 0 & \text{otherwise} \end{cases}$$

Table 4.2: Activated features on a training set example.

Feature Name	Num of occurences	Active on sequence
Basic Helix A	3	CAAF, DFAE
Basic Helix C	1	CDED
Basic Non-Helix C	1	CAAF
Basic Non-Helix D	3	CDED, DFAE
Basic Non-Helix E	2	CDED, DFAE
Basic Helix F	2	CAAF, DFAE
Hydrophobic Helix	6	CAAF, CDED, DFAE
Hydrophobic Non-Helix	1	CAAF
Polar Helix	1	CDED
Polar Non-Helix	6	CAAF, CDED, DFAE

We describe the relationship among the observation and the two possible structures, helix membrane/non-helix membrane as a feature (as described in the previous Section).

We train the CRFs model using the following training set:

```
CAAF
0111

CDED
1000

DFAE
0110
```

Our goal is to predict the labels of the sequence *EAFD*.

Table 4.2 shows the full list of activated features on the given training set.

In our training set 10 features are activated in total. After the model is trained, the CRFs will determine the parameters $\lambda = (\lambda_1, \lambda_2 ... \lambda_{10})$ assigned to each feature which maximize the log-likelihood of the model. Table 4.3 shows the values of

Table 4.3: Trained feature parameters.

Feature Name	Parameter value
Basic Helix A	1.7859
Basic Helix C	5.5170
Basic Non-Helix C	-5.5170
Basic Non-Helix D	1.7859
Basic Non-Helix E	1.5391
Basic Helix F	1.5391
Hydrophobic Helix	3.3251
Hydrophobic Non-Helix	-5.5170E-8
Polar Helix	5.5170E-8
Polar Non-Helix	3.3251

these feature parameters after they were calculated.

After calculating the feature parameters, the Viterbi algorithm is applied for labelling the test sequences. Now we will label the sequence *EAFD*. The first letter in the sequence, *E*, is a polar residue. From Table 4.3 we can calculate the total score of *E* by assigning it with a helical or a non-helical label. For assigning a helical label, the total score of *E* is *Polar Helix* ≈ 0, while for assigning a non-helical label, the total score is
Polar Non-Helix + Basic Non-Helix E = 4.8642. Similarly, the total score of each one of the letters assembling the sequence *EAFD* is calculated for each label. Table 4.4 summarizes the steps of calculating the sequence labels. Finally the

Table 4.4: Calculating the labels of the sequence *EAFD*.

Letter	Score of label 0	Score of label 1	Final score	Assigned Label
E	4.8642	0	4.8642	0
A	4.8642	9.9752	9.9752	1
F	9.9752	14.8394	14.8394	1
D	19.9504	14.8394	19.9504	0

algorithm assigns for each step these labels which yield the highest score. In this example *EAFD* is assigned with the predicted label sequence of *0110*.

4.9 Summary

Conditional Random Fields (CRFs) are a probabilistic framework from the family of conditional models for labelling sequential data. CRFs have a single exponential undirected graphical model for the joint probability distribution of the entire sequence of labels given the observation sequence. The conditional property of CRFs in which a sequence is labelled given its observation, and the fact that CRFs are undirected, make CRFs suitable for a seamless integration of biological domain knowledge into the model. CRFs models are heavily reliant on the maximum entropy theory. The main principal is to create a model that satisfies all known constraints but treat the unknown constraints uniformly. By satisfying a constraint we actually increase the likelihood of the global configuration. In CRFs, no feature-independence assumption is made, and several dependent overlapping features are allowed. During the training process, the model assigns each feature with a weight which is calculated using feature parameter estimation.

The parameter estimation problem is to determine each feature's parameter from the training data, subject to the constraint that the model distribution is equal the empirical distribution. This probability distribution as a function of the parameter is known as the *likelihood*. In maximum likelihood training, the goal is to maximize the logarithm of the likelihood, known as the log-likelihood. This training can be implemented using different techniques such as iterative scaling algorithm, Newton's methods, Quasi-Newton methods, and limited memory BFGS method.

The most important aspect of specifying the model is selecting a set of features that captures the relationships among the observation and the label sequences. In this thesis we have created 8 different types of features. Integrating these features into CRFs and reporting our experiment results are will be discussed in the next Chapter.

Chapter 5

Experiments, Evaluation and Analysis

In this Chapter we report on the experiments that we have carried out to test the efficacy of the CRF model for the transmembrane helix prediction problem. The CRF model was introduced and derived in Chapter 4 and the transmembrane helix prediction problem was described in Chapter 3.

We have carried out four sets of experiments with the objective of investigating the CRF model from different perspectives.

The four set of experiments are:

1. A comparison of the different feature selection strategies and their effect on the prediction accuracy. The CRFs with the best set of features were then evaluated against twenty eight other models at the "Static Benchmarking" website [16].

2. The second experiment was carried out to test the hypothesis that the location of the transmembrane region is determined by the distribution of amino acids in the region rather than their ordering. For example, if FCD is an observation sub-sequence then the probability that FCD is labelled as trans-

membrane helix is the same as the probability that the *permutation* of FCD is labelled as a transmembrane helix.

3. The third experiment compares the CRFs with maximum entropy Markov models (MEMMs). In particular we are interested in understanding the effect of the undirected CRFs versus directed MEMMs on the prediction accuracy.

4. The fourth experiment applies CRFs to analyze a large and well known IMP complex, the Cytochrome x oxidase. The analysis obtained is compared with the result of a similar analysis, derived using a "wet lab" experiments and reported by Wallin et al. [46].

Before we describe the experiments, we briefly overview the data sets and the evaluation metrics used throughout this Chapter.

5.1 Data Set

The CRF model was trained on a data set consisting of a set of benchmark sequences with experimentally confirmed transmembrane regions compiled by Möller et al. [30]. [1] We have only included proteins with a high level of trust (assigned with transmembrane annotation trust level of A to C, as was suggested by Möller et al.) and which are significantly different, based on pairwise similarity clustering. The resulting set consists of 148 transmembrane protein sequences.

The model was tested by using the "Static benchmarking of membrane helix predictions" website hosted in Columbia University [16]. This is a two-step process. First, a data set consisting of 2246 observation sequences was downloaded and labelled by the CRF model. Second, the labelled sequences were then uploaded to the website which reported a comparative performance analysis with twenty eight other models. The experimental work flow is illustrated in Figure 5.1.

[1] The data set can be accessed via ftp://ftp.ebi.ac.uk/databases/testsets/transmembrane

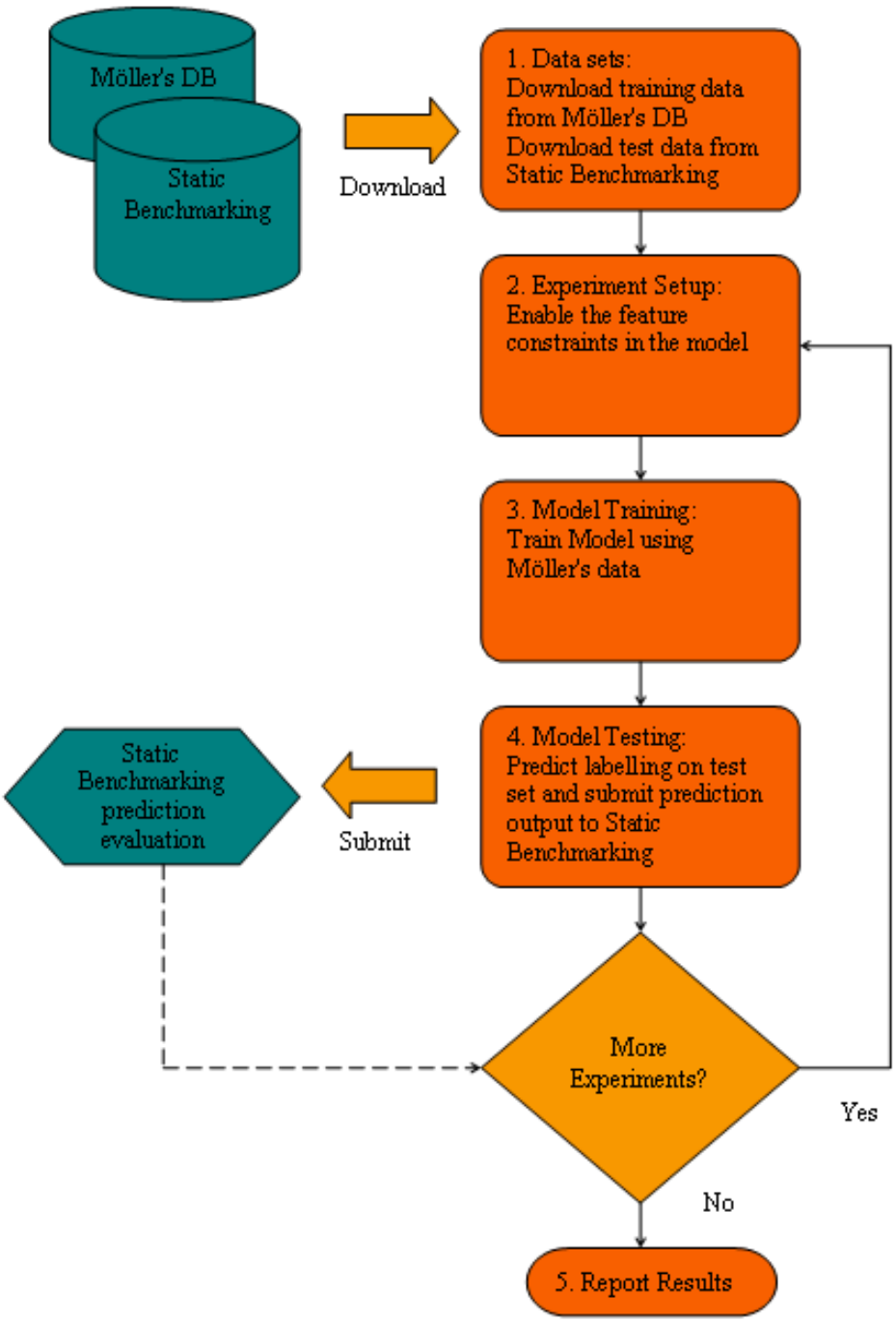


Figure 5.1: Transmembrane helix prediction experimental flow.

5.2 Prediction Metrics

There are two main approaches to rank a prediction model for membrane helices: per-residue and per-segment accuracy [5, 6].

5.2.1 Per-Residue Accuracy

In per-residue accuracy the predicted and actual labels are compared by residue. Let Ω_i be the sequence of all residues in protein $i = 1, \ldots, N_{prot}$.
Furthermore, let $\omega_{(i,j)} \in \Omega_i$ be the residue in location j in sequence Ω_i. For simplicity, we denote $\omega_{(i,j)}$ as ω. Let [2]

$$y(\omega) = \begin{cases} 1 & \text{if } \omega \text{ is a TMH residue} \\ 0 & \text{if } \omega \text{ is a NTMH residue} \end{cases}$$

Similarly, let

$$\tilde{y}(\omega) = \begin{cases} 1 & \text{if } \omega \text{ is predicted as a TMH residue} \\ 0 & \text{if } \omega \text{ is predicted as a NTMH residue} \end{cases}$$

Table 5.1 lists the metrics which capture per-residue accuracy. The symbols in the last column are from Chen et al., and are also used by the ”Static Benchmarking website” to report results.

5.2.2 Per-Segment Accuracy

The per-residue accuracy measures ignore the sequential contiguity of the transmembrane helical (TMH) regions. We also want to determine how accurately a method correctly predicts the location of a TMH region.

In order to predict the sequential contiguity of the TMH region we have used the per-segment accuracy metric suggested by Chen et al. [5]. It requires a minimal

[2] TMH = Transmembrane Helix, NTMH = Non-Transmembrane Helix

overlap of three residues between the two corresponding segments and does not allow the same helix to be counted twice. For example consider the following observed data and two possible prediction sequences: (0 = Non-Transmembrane Helix, 1 = Transmembrane Helix)

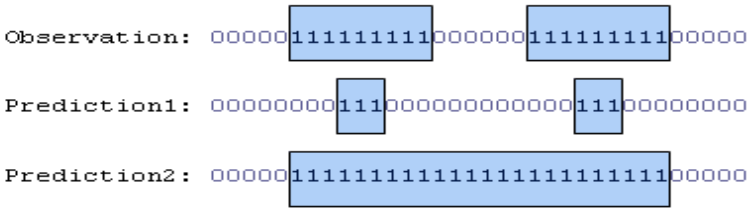


Figure 5.2: Per-segment accuracy example.

By using the per-segment metric, Prediction1 returns an accuracy of 100% (as it predicts two helices which are assigned with the two observation helices), while Prediction2 returns an accuracy of 50% (as it predicts one helix which is assigned only with the first Observation helix). Table 5.2 lists the metrics which capture per-segment accuracy.

Table 5.1: The five metrics used to measure per-residue accuracy.

Description	**Formula**	**Symbol**†
TMH Recall	$P(\tilde{y}(\omega) = 1 \mid y(\omega) = 1)$	$Q_{2T}^{\%obs}$
TMH Precision	$P(y(\omega) = 1 \mid \tilde{y}(\omega) = 1)$	$Q_{2T}^{\%prd}$
NTMH Recall	$P(\tilde{y}(\omega) = 0 \mid y(\omega) = 0)$	$Q_{2N}^{\%obs}$
NTMH Precision	$P(y(\omega) = 0 \mid \tilde{y}(\omega) = 0)$	$Q_{2N}^{\%prd}$
Residues correctly predicted	$\frac{1}{N_{prot}} \sum_{i=1}^{N_p rot} (P(\tilde{y}_i = 1, y_i = 1) + P(\tilde{y}_i = 0, y_i = 0)) \lvert\Omega_i\rvert$	Q_2

N_{prot} = Number of Proteins in the Data Set, TMH = Transmembrane Helix, NTMH = Non-Transmembrane Helix.

† These symbols are from Chen et al.[5]

Table 5.2: The three metrics used for measuring per-segment accuracy.

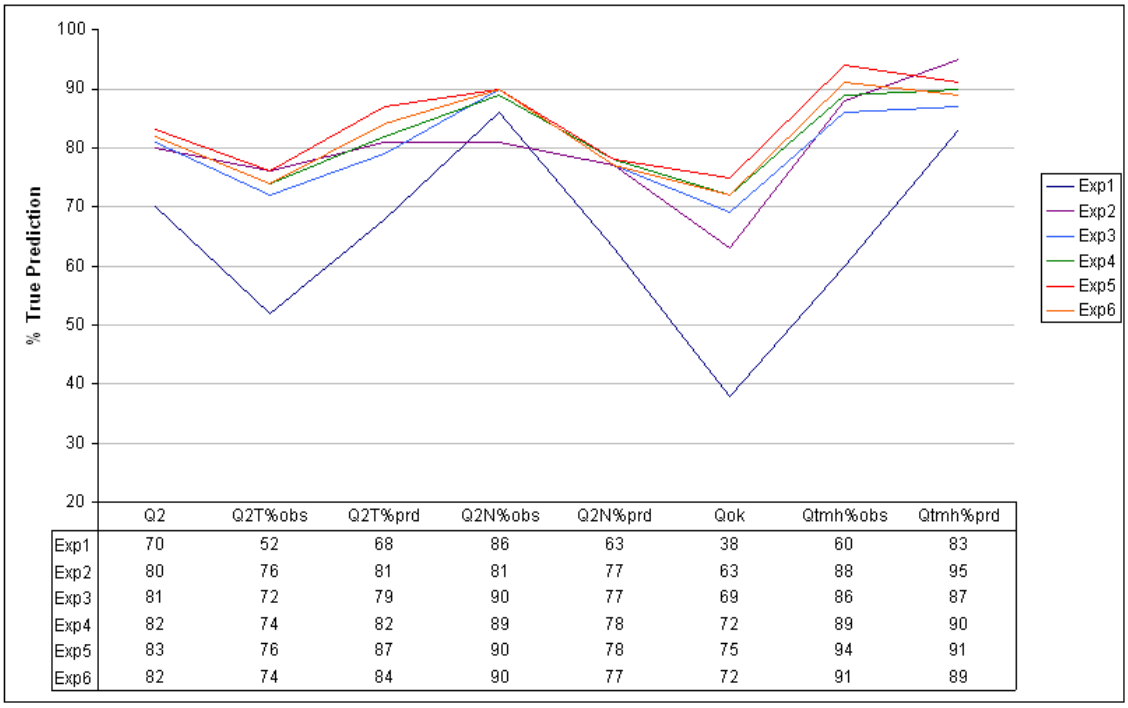
Description	**Symbol**†
All observed TMH which are correctly predicted	$Q_{tmh}^{\%obs}$
All predicted TMH which are correctly predicted	$Q_{tmh}^{\%prd}$
Proteins for which all TMH are correctly predicted	Q_{ok}

† These symbols are from Chen et al.[5]

5.3 Results and Analysis

5.3.1 Transmembrane Helix Prediction

The results of the CRFs model in predicting the TMH regions are shown in Figure 5.3. These results were obtained from the ”Static benchmarking of membrane helix predictions” website [16]. 6 different feature combinations were used which are described in Table 5.3. A detailed description of each of the features was given in Section 5.3.3.



	Q2	Q2T%obs	Q2T%prd	Q2N%obs	Q2N%prd	Qok	Qtmh%obs	Qtmh%prd
Exp1	70	52	68	86	63	38	60	83
Exp2	80	76	81	81	77	63	88	95
Exp3	81	72	79	90	77	69	86	87
Exp4	82	74	82	89	78	72	89	90
Exp5	83	76	87	90	78	75	94	91
Exp6	82	74	84	90	77	72	91	89

Figure 5.3: Prediction score for selected feature combinations.

Table 5.3: Enabled and disabled feature combination.

Experiment No.	Basic	Properties	Single	Double	Single Shuffled	Double Shuffled	Border	No. of Active Features
1	+	+	-	-	-	-	-	68
2	+	-	+2	+1	-	-	-	4634
3	-	+	+5	+1	-	-	-	591677
4	+	+	+3	+1	+3	+1	-	171079
5	+	+	+5	+3	+6	+3	-	931724
6	+	+	+5	+3	+6	+3	+	938292

Basic = Basic Amino Acid Features, *Properties* = Amino Acid Property Features, *Single* = Single Side Neighboring Amino Acid Features (with 1 to 5 neighbors from left or right), *Double* = Double Side Neighboring Amino Acid Features (with 1 to 3 neighbors from both sides), *Single Shuffled* = Single Side Shuffled Neighboring Amino Acid Features (with 1 to 6 neighbors from left or right), *Double Shuffled* = Double Side Shuffled Neighboring Amino Acid Features (with 1 to 3 neighbors from both sides). All experiments are using *Start*, *Edge*, and *End* features by default, which are essential for defining the proteins from the data sets as sequences.

As expected, the models's performance is crucially dependent upon feature selection. The two important scores, Q_2 (per-residue) and Q_{ok} (per-segment), as defined in Tables 5.1 and 5.2, are influenced most by incorporating single side and double side neighboring features. Adding Single Side Neighboring Feature of two neighbors and Double Side Neighboring Feature of one neighbor, increase Q_2 from 70% to 80% and Q_{ok} from 38% to 63% (Experiment 1 vs. 2). Increasing the number of Single side and Double side neighbors has a small effect on Q_2 score, but a dramatic effect on Q_{ok}. This can be explained by the definition of Q_{ok} which is a per-segment metric, and is more dependent on the neighboring amino acids than Q_2.

Enabling as many features as possible is not a guarantee for a better prediction score, as can be seen from the results of Experiment 5 and 6.

Experiment 5 (henceforth referred as "the CRFs model") outperforms the other experiments in most categories and was selected for comparison with other prediction models.

We submitted the result of the CRFs model on the test data set to the "Static benchmarking of membrane helix predictions" website and obtained a comparative ranking against other available methods. This is shown in Table 5.4, and indicates that our model performed well both on per-residue and per-segment predictions. The CRFs model achieved the highest score among all the 29 methods in the overall percentage of residues correctly predicted (Q_2) with an accuracy of 83%. On the per-segment test, the CRFs model obtained Q_{ok} score of 75%. This is the fourth highest score and means that all the transmembrane helices were correctly predicted for every three out of four proteins. Also the total percentage of correctly predicted helices was 94% with a precision of 91%.

Table 5.4: Prediction score comparison between 29 methods.

Per-Residue						Per-Segment		
Methods	Q_2	$Q_{2T}^{\%obs}$	$Q_{2T}^{\%prd}$	$Q_{2N}^{\%obs}$	$Q_{2N}^{\%prd}$	Q_{ok}	$Q_{tmh}^{\%obs}$	$Q_{tmh}^{\%prd}$
CRFs	**83**	**76**	**87**	**90**	**78**	**75**	**94**	**91**
TMHMM1	80	68	81	89	72	71	90	90
PHDpsihtm08	80	76	83	86	80	84	99	98
HMMTOP2	80	69	89	88	71	83	99	99
PHDhtm08	78	76	82	84	79	64	77	76
PHDhtm07	78	76	82	84	79	69	83	81
TopPred2	77	64	83	90	69	75	90	90
PRED-TMR	76	58	85	94	66	61	84	90
SOSUI	75	66	74	80	69	71	88	86
DAS	72	48	94	96	62	79	99	96
Ben-Tal	72	53	80	95	63	60	79	89
WW	71	71	72	67	67	54	95	91
GES	71	74	72	66	69	64	97	90
Eisenberg	69	77	68	57	68	58	95	89
KD	67	79	66	52	67	65	94	89
Sweet	63	83	60	43	69	43	90	83
Wolfenden	62	28	56	97	56	28	43	62
Hopp-Woods	62	80	61	43	67	56	93	86
Heijne	61	85	58	34	64	45	93	82
Nakashima	60	84	58	36	63	39	88	83
Av-Cid	60	83	58	39	72	52	93	83
Levitt	59	80	58	38	67	48	91	84
Roseman	58	83	58	34	66	52	94	83
A-Cid	58	80	56	37	66	47	95	83
EM	57	85	55	28	64	31	92	77
Radzicka	56	85	55	26	63	40	93	79
Fauchere	56	84	56	31	65	36	92	80
Lawson	55	85	55	27	66	33	86	79
Bull-Breese	55	85	55	27	66	45	92	82

Taken from the ”Static benchmarking of membrane helix predictions” application written by Kernytsky et al. from University of Columbia, after submitting our CRFs prediction results (sorted by Q_2)[16].

5.3.2 The Effect of Local Residue Distribution on Transmembrane Regions

We are interested in testing the hypothesis that the location of the transmembrane regions is determined by the composition of the amino acids in the transmembrane regions rather than the sequencing. In order to test this hypothesis, we have carried out two experiments with different combinations of features turned on as shown in Table 5.5. In experiment 1 we have used Single and Double Side Neighboring Amino Acid Features and in experiment 2 we have used Single and Double Side Shuffled Neighboring Amino Acid Features (see Section 5.3.3 for an explanation of these features). We have set the null hypothesis to be that the mean of performance scores of the first experiment using Single and Double Side features is equal to using Single and Double Side Shuffled features. We can formulate the hypothesis as:

$$H_0 : \mu^d = \mu^{Single+Double} - \mu^{Single+DoubleShuffled} = 0 \tag{5.1}$$

The prediction results of the two experiments are shown in Table 5.6. A t-test on the results with 99% confidence resulted in a p-value = 0.8035 with t = -0.2585 ,a confidence interval of [-5.452159, 4.702159], and mean of the differences = -0.375. These indicates that there is not enough evidence to reject the null hypothesis that the means are equal. Therefore, there is no evidence that the transmembrane regions are determined by a specific sequencing of the amino acids rather than their composition.

Table 5.5: The two different feature combinations used to test the effect of local residue distribution on TMH prediction.

Experiment No.	Basic	Properties	Single	Double	Single Shuffled	Double Shuffled	Border	No. of Active Features
1	+	+	+5	+3	-	-	-	803260
2	+	+	-	-	+5	+3	-	88594

Table 5.6: Transmembrane prediction results using two different feature combinations.

Per-Residue						Per-Segment		
Methods	Q_2	$Q_{2T}^{\%obs}$	$Q_{2T}^{\%prd}$	$Q_{2N}^{\%obs}$	$Q_{2N}^{\%prd}$	Q_{ok}	$Q_{tmh}^{\%obs}$	$Q_{tmh}^{\%prd}$
1	83	72	84	93	78	75	87	91
2	83	75	86	89	77	75	92	83

Taken from the ”Static benchmarking of membrane helix predictions” application written by Kernytsky et al. from University of Columbia, after submitting the prediction results of both experiments (sorted by Q_2)[16].

5.3.3 Comparison of CRFs and MEMMs

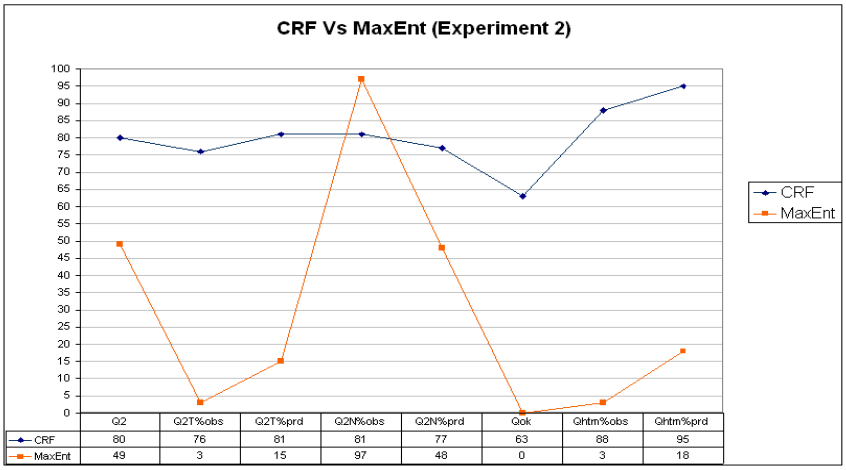
We now present evidence which suggests that CRFs outperform MEMMs for TMH prediction using the set of features which we have introduced on . We have repeated four out of the six experiments described in Section 5.3.1, using the MEMMs prediction model. Note that Experiment 6 could not be performed using MEMMs since it includes a Border feature which is only applicable on undirected graphical models, and therefore was excluded from our comparison. We have also excluded Experiment 1, as MEMMs have completely failed to predict any transmembrane regions using the set of features defined in this experiment!

In order to do the comparison we will test the null hypothesis that the difference of the performance metrics between CRFs and MEMMs is less than or equal to zero. This can be stated as:

$$H_0 : \mu^d = \mu^{CRFs} - \mu^{MEMMs} \leq 0 \tag{5.2}$$

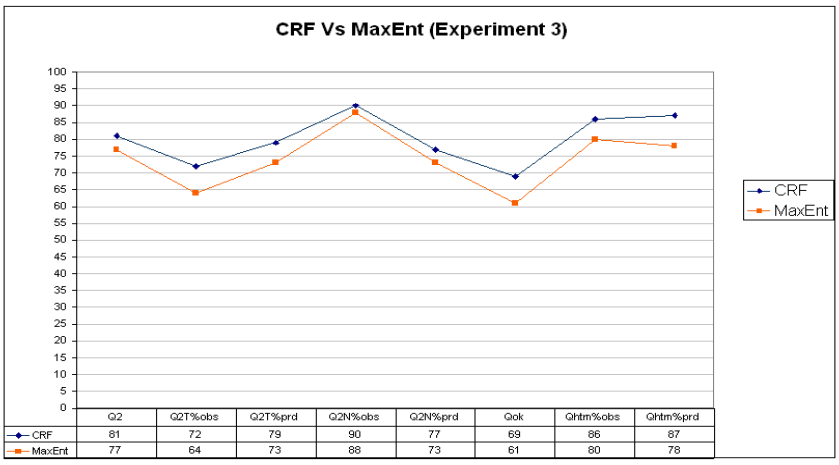
$$d_1 = Q_2^{CRFs} - Q_2^{MEMMs}, ..., d_8 = Q_{tmh}^{\%prd^{CRFs}} - Q_{tmh}^{\%prd^{MEMMs}}$$

The results of Experiments 2 to 5 are displayed in Figures 5.4 to 5.7 and summarized in the Table 5.7:



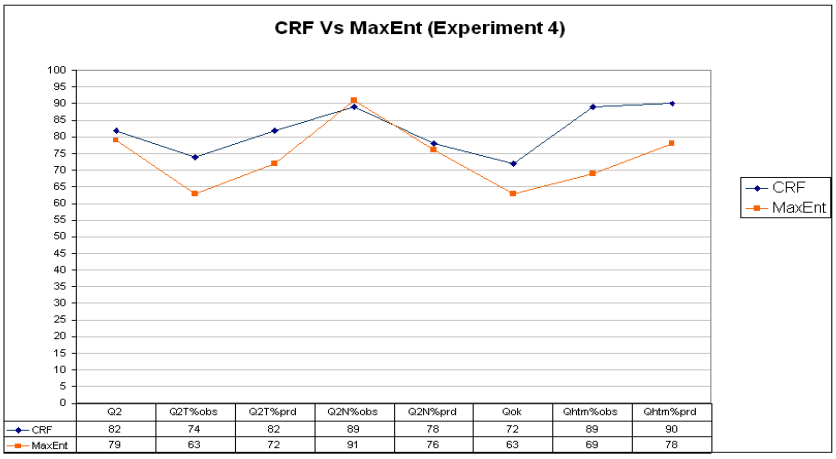
	Q2	Q2T%obs	Q2T%prd	Q2N%obs	Q2N%prd	Qok	Qhtm%obs	Qhtm%prd
CRF	80	76	81	81	77	63	88	95
MaxEnt	49	3	15	97	48	0	3	18

Figure 5.4: CRFs vs. MEMMs performance test on Experiment 2.



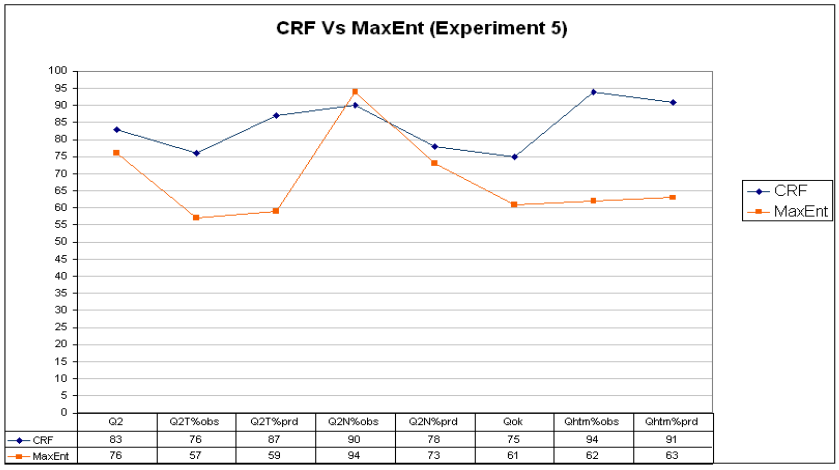
	Q2	Q2T%obs	Q2T%prd	Q2N%obs	Q2N%prd	Qok	Qhtm%obs	Qhtm%prd
CRF	81	72	79	90	77	69	86	87
MaxEnt	77	64	73	88	73	61	80	78

Figure 5.5: CRFs vs. MEMMs performance test on Experiment 3.



	Q2	Q2T%obs	Q2T%prd	Q2N%obs	Q2N%prd	Qok	Qhtm%obs	Qhtm%prd
CRF	82	74	82	89	78	72	89	90
MaxEnt	79	63	72	91	76	63	69	78

Figure 5.6: CRFs vs. MEMMs performance test on Experiment 4.



	Q2	Q2T%obs	Q2T%prd	Q2N%obs	Q2N%prd	Qok	Qhtm%obs	Qhtm%prd
CRF	83	76	87	90	78	75	94	91
MaxEnt	76	57	59	94	73	61	62	63

Figure 5.7: CRFs vs. MEMMs performance test on Experiment 5.

A t-test on the experimental results taken from Table 5.7 with 99% confidence on each pair of experiments clearly shows evidence against the null hypothesis defined above (5.2). The p-values that we obtained on the t-tests on Experiments 2,3,4 and 5 are 0.001893, 0.01007, 0.00636, and 0.004705 respectively. These results suggest that we accept the alternative hypothesis that the difference between the performance metrics of CRFs and MEMMs is greater than zero, which support the conjecture we made in Section 3.5.

Table 5.7: Prediction score comparison between CRFs and MEMMs.

Per-Residue						Per-Segment		
Experiment	Q_2	$Q_{2T}^{\%obs}$	$Q_{2T}^{\%prd}$	$Q_{2N}^{\%obs}$	$Q_{2N}^{\%prd}$	Q_{ok}	$Q_{tmh}^{\%obs}$	$Q_{tmh}^{\%prd}$
2^{CRFs}	80	76	81	81	77	63	88	95
2^{MEMMs}	49	3	15	97	48	0	3	18
3^{CRFs}	81	72	79	90	77	69	86	87
3^{MEMMs}	80	68	81	89	72	61	80	78
4^{CRFs}	82	74	82	89	78	72	89	90
4^{MEMMs}	79	63	72	91	76	63	69	78
5^{CRFs}	83	76	87	90	78	75	94	91
5^{MEMMs}	76	57	59	94	73	61	62	63

Taken from the "Static benchmarking of membrane helix predictions" application written by Kernytsky et al. from University of Columbia, after submitting our CRFs and MEMMS prediction results (sorted by Q_2)[16].

The superior performance of CRFs over MEMMs can be explained as follows: MEMMs use a per-state conditional model to calculate the probability of entering the current state given the previous state, generating a directed graphical model. In contrast, the CRFs use a single global exponential model to calculate the entire sequence of states given the observation sequence, generating an undirected graphical model [21]. Since CRFs are based on global model, they can overcome the label bias problem, as defined in Section 3.5. The label bias problem is most dramatically illustrated in Experiment 1 where MEMMs completely fail to predict any transmembrane region, while CRFs manage to predict some of these regions.

In Experiment 1 we have defined two types of features: Basic and Property features. Both feature types are based on the probability of the current amino acid (or the property of the current amino acid) to appear with a given label. Table 3.2 shows the conditional probability distribution of the amino acids and the labels. We can easily notice that $P(Y = 1|X) \leq 0.5$ where X ranges over all amino acids. This implies that by itself none of the amino acids are likely to appear inside a transmembrane region than outside. Furthermore, the probability

of $p(y_i = 1|y_{i-1} = 0, X) \approx p(y_i = 1|y_{i-1} = 0)$ for all amino acids X, as shown in Figure 5.8. This causes the transition from state 0 to state 1 to completely ignore the observation data in the MEMMs model.

CRFs, however, do take into account the fact that assigning an amino acid to a transmembrane region may improve the overall score of the entire sequence, as demonstrated in the CRFs results of Experiment 1.

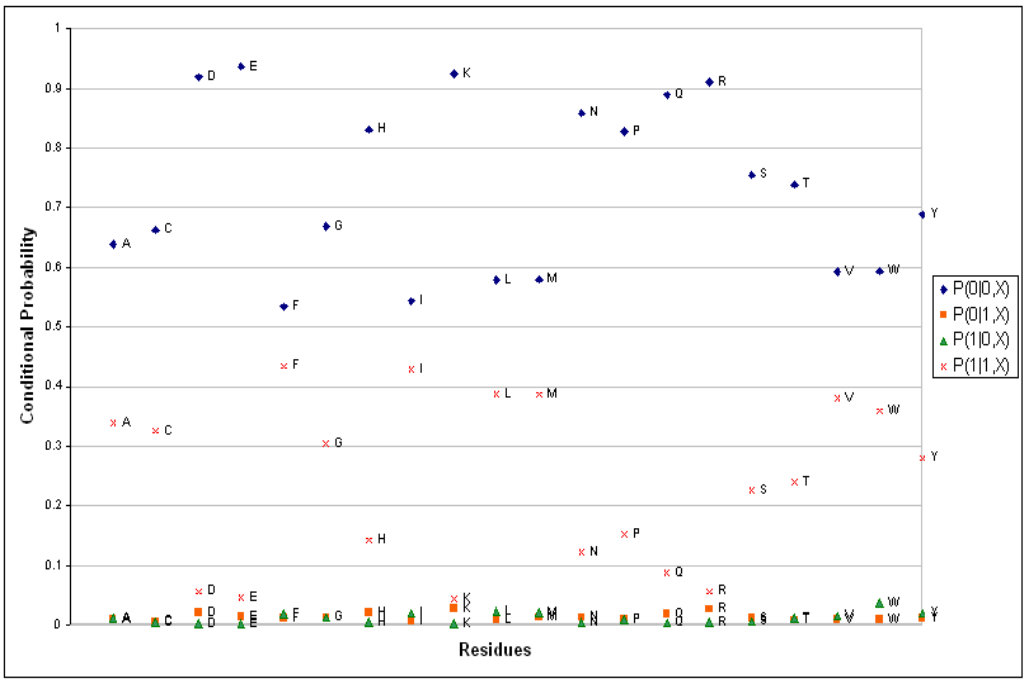


Figure 5.8: The conditional probability $P(y_i|y_{i-1}, x)$ of residues for different transition states.

When more complex features are used, the effect of the label bias problem is ameliorated, but the differences between the performance of CRFs and MEMMs are still significant, as presented earlier. In Figure 5.9 we present several protein sequences that were predicted by CRFs and MEMMs using the set of features defined in Experiment 5. Notice the prediction similarities and dissimilarities of the two models. While the *Atpl Ecoli* and the *Glpa Human* proteins were predicted identically by both models, MEMMs have missed a segment on the *Kcsa Strli*

protein and joined two segments on *Bacr Halha* protein. In contrast, CRFs did detect all segments for these proteins, with a minor segment shift on *Kcsa Strli* and *Bacr Halha* proteins.

Figure 5.9: CRFs vs. MEMMs prediction output.

5.3.4 Cytochrome c oxidase Protein Analysis

The Cytochrome c oxidase protein has one of the most complex structures known to date with a total of 28 transmembrane helices [46]. A detailed "wet lab" analysis of this complex, in terms of residue distribution and amino acid properties has been reported by Wallin et al. [46] (henceforth referred as the observed results). We recreate the results using the labels predicted by the CRFs model using the features defined in Experiment 5 (see Table 5.3).

We begin by predicting the distribution of individual residue type in the central membrane domain ±10Å. The predicted and the observed results are shown in Figure 5.11. We can easily notice the high similarity between the observed and the predicted frequencies. We conducted a paired t-test on the observed and predicted data, testing the hypothesis that the mean of the observed and the predicted frequencies are equal. Using 99% confidence interval we got p-value of 0.35% indicating that there is no evidence that the mean of the observed and predicted frequencies are different.

Wallin et al. [46] also analyzed the frequency of the residues around the transmembrane helices based on the division of the amino acids into three groups: hydrophobic, polar and charged. We carried out a similar analysis using the CRFs model and the results are shown in Figure 5.12. The concentration of hydrophobic residues around the central helix (±13 residues around 0) is in the frequency range between 20 to 35% (average of 24%). It is interesting to note that the frequency distribution of the polar residues is a virtual mirror image of the hydrophobic frequency along the central helix (average of 11.5%). The charged residues tend to appear with an average frequency of 6% outside the helix membrane and 3.5% inside. These results are compatible with physical experiments which show that highly polar and charged amino acids are energetically unfavorable inside the membrane, with the hydrophobic core of the cell membrane filling between the polar head groups. Recent studies by Hessa et al.[13] using a biological assay of membrane segments insertion suggest that not only the degree of

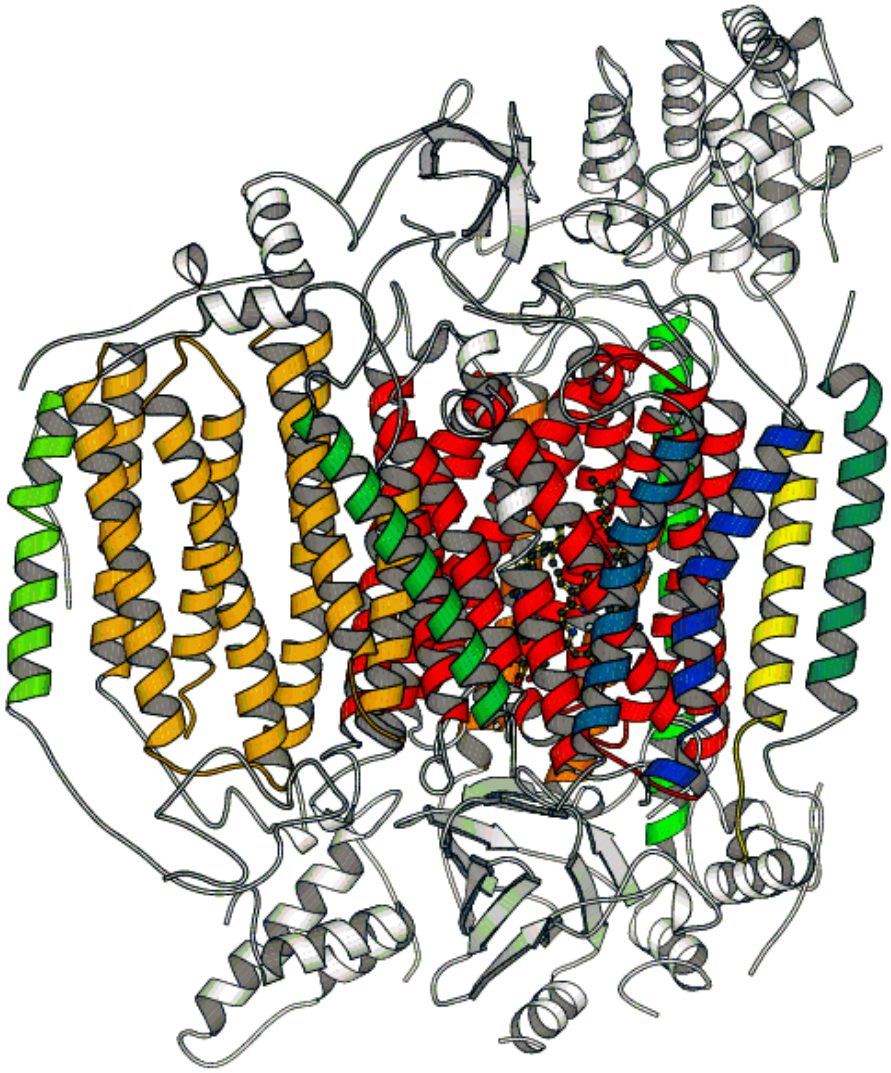


Figure 5.10: The crystal structure of a subunit of mitochondrial Cytochrome c oxidase [41]. The plane of the membrane is approximately horizontal showing the 28 transmembrane spanning helices with the intra-cellular space at the bottom of the figure. Substantive extracellular domains of the protein are also to be seen outside the membrane region. Figure prepared with MOLSCRIPT [20].

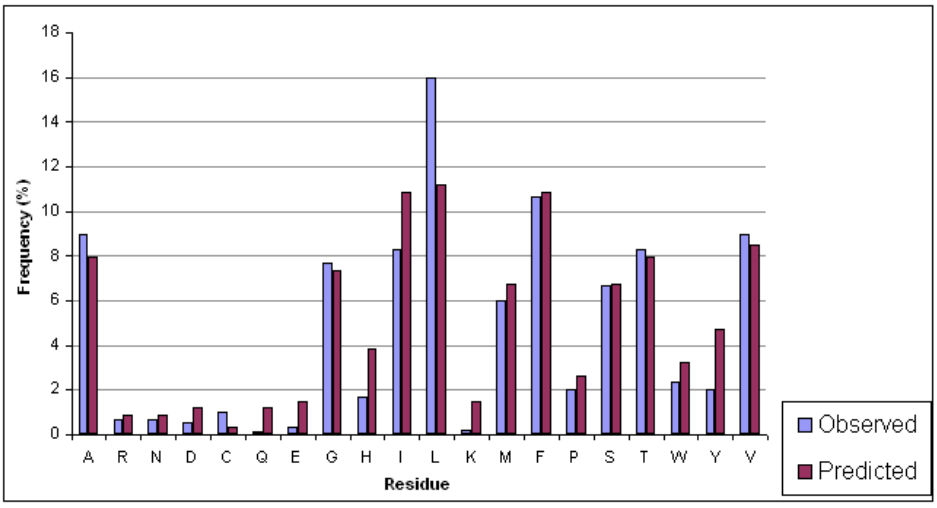


Figure 5.11: Predicted vs. observed residue frequencies in Cytochrome c oxidase central membrane domain ± 10 Å region.

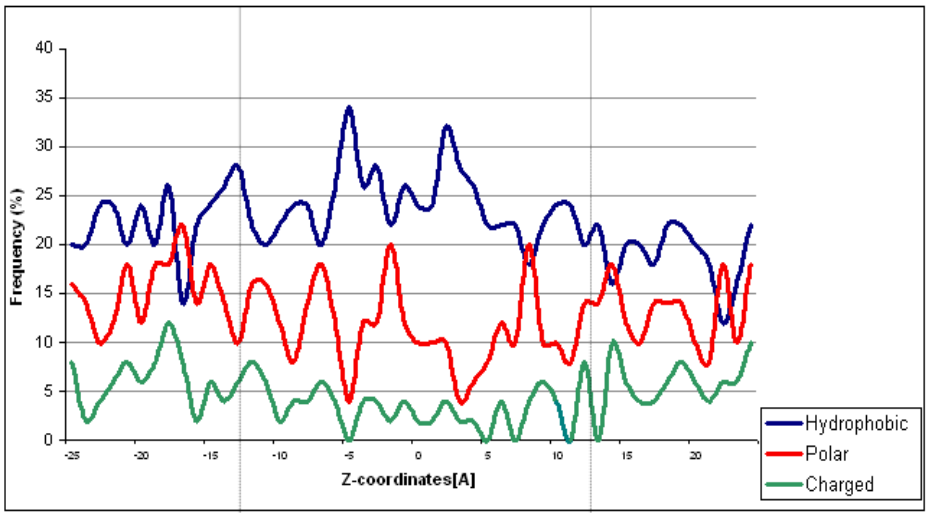


Figure 5.12: Properties of the predicted residues the in central membrane domain ± 25 Å region.

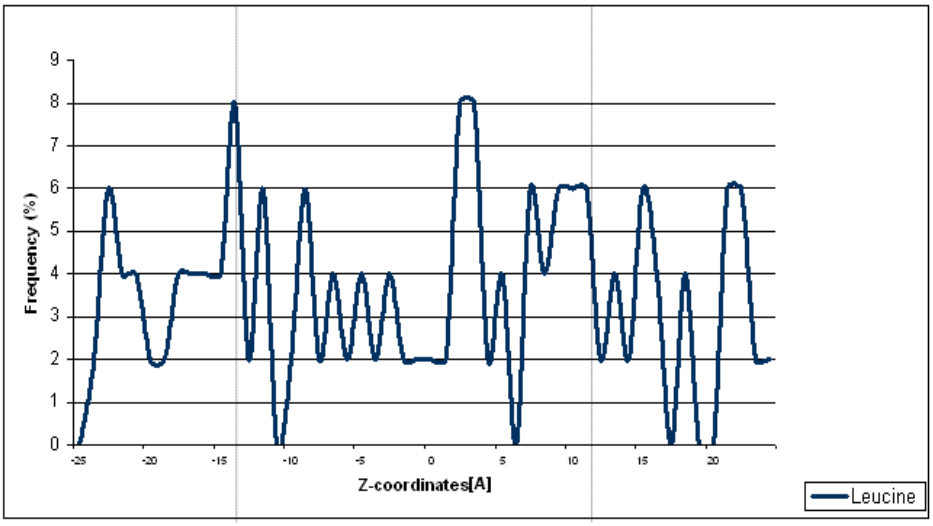


Figure 5.13: Frequency of leucine in the central membrane domain ± 25 Å region.

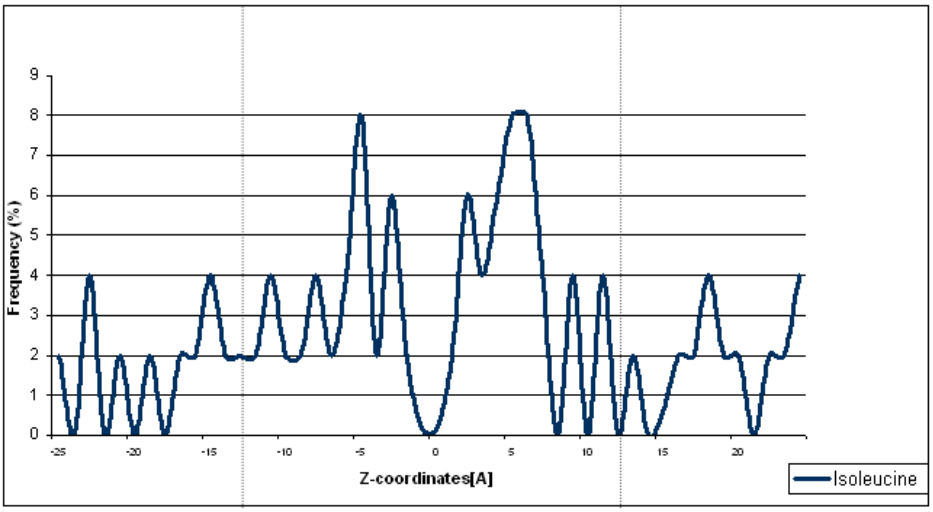


Figure 5.14: Frequency of isoleucine in the central membrane domain ± 25 Å region.

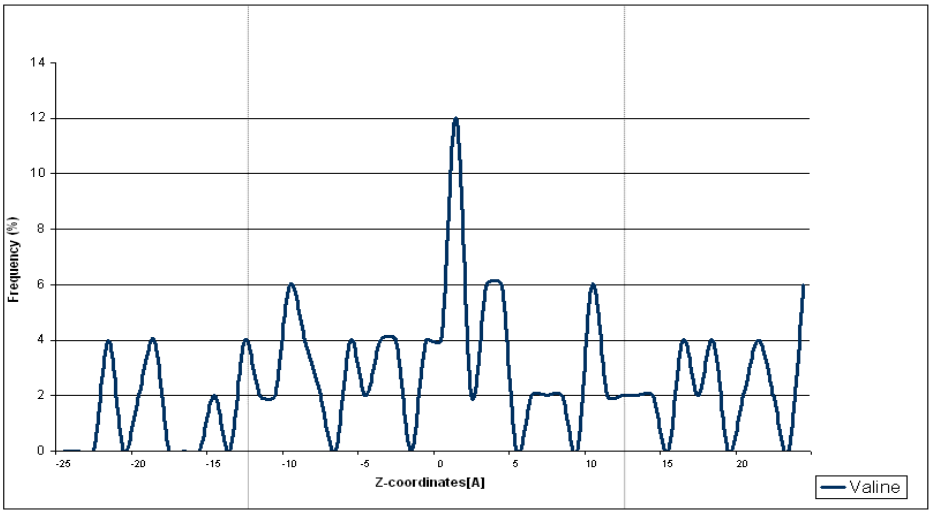


Figure 5.15: Frequency of valine in the central membrane domain ± 25 Å region.

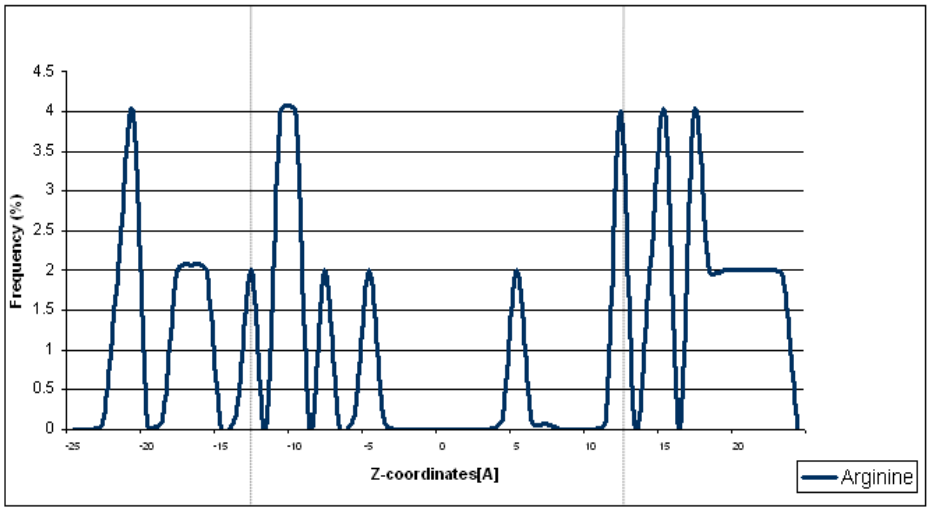


Figure 5.16: Frequency of arginine in the central membrane domain ± 25 Å region.

hydrophobicity is important in structuring transmembrane helices but also the energetic stability of the helices, defined by the sum of individual contribution from each amino acid. The authors have demonstrated that the contribution to the total apparent free energy depends strongly on the position of each residue within the helix [13]. Generally the more the polar and charged amino acids are close to the membrane center, the larger the energetic penalty generally is. The accepted explanation is that electrostatic forces close to the polar head group and beyond the membrane help to stabilize the polar and charged groups in amino acids.

In support of this conjecture, we predicted the tendency of arginine (both polar and charged residue) around the transmembrane helix. We found that the tendency of arginine favorable to be located outside the membrane as demonstrated in Figure 5.16.

In contrast, we predicted the frequency around the membrane helix of isoleucine, leucine, and valine as an example of hydrophobic amino acids which according to observed experiments are favorable to appear in the center of the membrane. The results are demonstrated in Figures 5.13- 5.14.

5.3.4.1 Approach

On the Cytochrome c oxidase experiments, we collected the distribution of individual residue type in the central membrane domain assuming that the central is on average ± 9 residues from the transmembrane border. On Wallin's work, he has calculated the distribution on a region of ± 10Å [46], which we calculated using SwissPdbView [12], and found that on average equals to 9 residues long. Wallin also referred his distribution figures to three different profiles: buried, intermediate and fully exposed residues. We compared our prediction results to the average values of the three profiles, since the sequence input that we used does not distinguish between these profiles.

The CRFs model was trained using data set consists of a set of benchmark se-

quences with experimentally confirmed transmembrane regions, which are significantly different, based on pairwise similarity clustering [30]. All occurrences of Cytochrome c oxidase sequences were removed from the training set to assure that the training set and test set are not overlapping.

Conducting t-test on the hypothesis that the difference in means between the predicted and the observed residue frequencies are equal to 0, yields results of t = 0.9653, df = 19 and p-value = 0.3465, with 99% confidence interval of [-0.0067, 0.0136]. The sample estimates mean of the differences is 0.00343.

5.4 Summary

Structural determination of integral membrane proteins can be problematic due to difficulties in obtaining sufficient amounts of sample. Therefore good prediction methods of membrane structures are highly valued.

In this thesis we have used Conditional Random Fields (CRFs) to predict the location of transmembrane helices in membrane proteins. Our results look very promising compared to currently available methods, as the CRFs model outperforms the other methods on important metrics. The CRFs model achieved the highest score among all 29 methods in the overall percentage of residues predicted correctly in both transmembrane and non-transmembrane helices (Q_2) with 83% of true prediction. On the per-segment test, our model achieved the fourth highest score with 75% of proteins for which all transmembrane helices were predicted correctly (Q_{ok}).

In the second set of experiments, we have shown the hypothesis that the location of the transmembrane regions are determined by the difference in the amino acid distribution in various structural parts of the protein, rather than a specific sequence of amino acid in these parts.

In the third set of experiments, we have provided some evidence that CRFs outperform MEMMs for transmembrane helix prediction and help overcome the label bias problem.

In the fourth and final set of experiments we have used CRFs to analyze the architecture of the protein complex, Cytochrome C Oxidase. We were able to recreate the results of amino acid distribution around the membrane regions similar to those obtained in a ”wet lab” experiments. These clearly demonstrates the usefulness of the CRFs model for the transmembrane helix prediction problem.

5.5 Conclusions

In this thesis we have used Conditional Random Fields (CRFs) as a sequential classifier to predict the location of transmembrane helical regions in membrane proteins. CRFs allow for a seamless and principled integration of biological domain knowledge into the model and provide several advantages over other approaches. We compared our approach with twenty eight other methods available through the "Statistic benchmarking of membrane helix predictions" [16]. The CRFs model (with our choice of features) received the highest score in the overall percentage of residues correctly predicted. We have also compared CRFs and MEMMs, and shown that CRFs are able to overcome the label bias problem from which MEMMs are known to suffer. Finally, we have used CRFs model to analyze the architecture of the protein complex, Cytochrome c oxidase. We were able to recreate the wet lab results obtained by [46].

At the time when the research commenced in early 2004, most of the applications of CRFs were in the natural language processing domain. The literature was lacking applications of CRFs in bioinformatics domain, and our work was novel [26]. Today, the use of CRFs in bioinformatics is growing and successful applications, such as Gene prediction using CRFs, have appeared in the literature [7]. There are still many open problems where CRFs can be applied in bioinformatics.

We believe that CRFs have a great potential in bioinformatics and their use will cut down the number of wet experiments leading to a significant saving in research time and resources.

Bibliography

[1] Alison Abbott. Functional genomics: Structures by numbers. *Nature*, 408(6809):130–132, 2000.

[2] Adam Berger. The improved iterative scaling algorithm: A gentle introduction. Technical report, Carnegie Mellon University, 1997.

[3] Helen M. Berman, John Westbrook, Zukang Feng, Gary Gillil, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, and Philip E. Bourne. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.

[4] Eugen C. Buehler and Lyle H. Ungar. Maximum entropy methods for biological sequence modeling. In *Workshop on Data Mining in Bioinformatics*, pages 60–64, 2001.

[5] Chien Peter Chen, Andrew Kernytsky, and Burkhard Rost. Transmembrane helix predictions revisited. *Protein Science*, 11:2774–2791, 2002.

[6] Chien Peter Chen and Burkhard Rost. State-of-the-art in membrane protein prediction. *Applied Bioinformatics*, 1:21–35, 2002.

[7] Aron Culotta, David Kulp, and Andrew McCallum. Gene prediction with conditional random fields. Technical report, University of Massachusetts, 2005.

[8] J. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43:1470–1480, 1972.

[9] Robert Dodier. RISO: distributed belief networks. Sourceforge, June 2004. [Online] Available http://sourceforge.net/projects/riso.

[10] David G. Forney. The viterbi algorithm. In *Proceedings of the IEEE*, volume 61, pages 268–278, March, 1973.

[11] Reginald H. Garrett and Charles M. Grisham. *Biochemistry*. Saunders College Publishing, third edition, 2005.

[12] Nicolas Guex and Manuel C. Peitsch. Swiss-model and the swiss-pdbviewer: An environment for comparative protein modeling. *Electrophoresis*, 18:2714–2723, 1997. [Online] Available http://www.expasy.org/spdbv/.

[13] Tara Hessa, Hyun Kim, Karl Bihlmar, Carolina Lundin, Jorrit Boekel, Helena Andersson, IngMarie Nilsson, Stephen H. White, and Gunnar von Heijne. Recognition of transmembrane helices by the endoplasmic reticulum translocon. *Nature*, 433:377–381, 2005.

[14] Wolfgang Hoschek. The Colt distribution: Open source libraries for high performance scientific and technical computing in JAVA, November 2002. [Online] Available http://hoschek.home.cern.ch/hoschek/colt/.

[15] Henry Jakubowski. Biochemistry online, July 2005. [Online] Available http://employees.csbsju.edu/hjakubowski/classes/ch331/bcintro/default.html.

[16] Andrew Kernytsky and Burkhard Rost. Static benchmarking of membrane helix predictions. Nucl Acids Res, 31 In press., 2003. [Online] Available http://cubic.bioc.columbia.edu/cgi-bin/var/kernytsky/tmh/advanced-process.cgi.

[17] Yohan Kim. Application of maximum entropy markov models on the protein secondary structure predictions. Technical report, University of California, 2001.

[18] Jeffery Klauda. Transport of molecules through the cell membrane

via membrane proteins, August 2005. [Online image] Available http://www.lobos.nih.gov/˜klauda/Images/lacY-pope.JPG.

[19] Dan Klein. The Stanford Classifier. The Stanford Natural Language Processing Group, 2003. [Online] Available http://nlp.stanford.edu/software/classifier.shtml.

[20] Per J. Kraulis. Molscript: A program to produce both detailed and schematic plots of protein structures. *J. Appl. Cryst.*, 24:946–950, 1991.

[21] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[22] John F. Leite, Andrew A. Amoscato, and Michael Cascio. Coupled proteolytic and mass spectrometry studies indicate a novel topology for the glycine receptor. *Biol. Chem.*, 275:13683–13689, 2000.

[23] Arthur M. Lesk. *Introduction to Protein Science Architecture, Function, and Genomics*. Oxford University Press, 2004.

[24] Stan Z. Li. *Markov Random Field Modeling in Computer Vision*. Springer-Verlag New York, 1995.

[25] Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[26] Lior Lukov, Sanjay Chawla, and W. Bret Church. Conditional random fields for transmembrane helix prediction. In *9th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, volume 3518, pages 155–161. Springer-Verlag, 2005.

[27] Robert Malouf. A comparison of algorithms for maximum entropy parameter estimation. Technical report, Alfa-Informatica, 2002.

[28] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical*

Natural Language Processing. The MIT Press, Cambridge, Massachusetts, 1999.

[29] Andrew McCallum. Efficiently inducing features of conditional random fields. In *Proc. 19th Conference on Uncertainty in Artificial Intelligence*, 2003.

[30] Steffen Moller, Evgenia V. Kriventseva, and Rolf Apweiler. A collection of well characterized integral membrane proteins. *Bioinformatics*, 16:1159–1160, 2000.

[31] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer-Verlag, 1999.

[32] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[33] Marie Petracek. Secondary structure, July 2005. [Online image] Available http://opbs.okstate.edu/˜petracek/4113%20link.html.

[34] Stephen Della Pietra, Vincent Della Pietra, and John Lafferty. Inducing features of random fields. In *IEEE Transactions Pattern Analysis and Machine Intelligence*, volume 19, pages 380–393, 1997.

[35] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. volume 77, pages 257–285. IEEE, 1989.

[36] Sunita Sarawagi. CRF project. Sourceforge, 2004. [Online] Available http://crf.sourceforge.net/.

[37] Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, volume 1, pages 134 – 141. Association for Computational Linguistics, Morristown, NJ, USA, 2003.

[38] Richard J. Simpson. *Proteins and Proteomics: A Laboratory Manual*. Cold Spring Harbor Laboratory Press, 2003.

[39] Michael J.E. Sternberg. *Protein Structure Prediction: A Practical Approach*. Number 170 in Practical Approach Series. Oxford University Press, 1996.

[40] Tomitake Tsukihara, Hiroshi Aoyama, Eiki Yamashita, Takashi Tomizaki, Hiroshi Yamaguchi, Kyoko Shinzawa-ltoh, Ryosuke Nakashima, Rieko Yaono, and Shinya Yoshikawa. Structures of metal sites of oxidised bovine heart cytochrome c oxidase at 2.8A. *Science*, 269:1069–1074, 1995.

[41] Tomitake Tsukihara, Hiroshi Aoyama, Eiki Yamashita, Takashi Tomizaki, Hiroshi Yamaguchi, Kyoko Shinzawa-ltoh, Ryosuke Nakashima, Rieko Yaono, and Shinya Yoshikawa. The whole structure of the 13-subunit oxidised cytochrome c oxidase at 2.8A. *Science*, 272:1136–1144, 1996.

[42] Gabor E. Tusnady and Istvan Simon. Principles governing amino acid composition of integral membrane prteins: Application to topology prediction. *J. Mol. Biol*, 283:489–506, 1998.

[43] Gabor E. Tusnady and Istvan Simon. The HMMTOP transmembrane topology prediction server. *Bioinformatics Application Note*, 17(9):849–850, 2001.

[44] Hanna M. Wallach. Efficient training of conditional random fields. Master's thesis, University of Edinburgh, 2002.

[45] Hanna M. Wallach. Conditional random fields: An introduction. Technical Report MS-CIS-04-21, University of Pennsylvania, 2004.

[46] Erik Wallin, Tomitake Tsukihara, Shinya Yoshikawa, Gunnar von Heinjne, and Arne Elofsson. Architecture of helix bundle membrane proteins: An analysis of cytochrome c oxidase from bovine mitochondria. *Protein Science*, 6:808–815, 1997.

[47] David M. Webster. *Protein Structure Prediction: Methods and Protocols*, volume 143 of *Methods in Molecular Biology*. Humana Press, 2000.

[48] Alan Wise, Katy Gearing, and Stephen Rees. Target validation of G-protein coupled receptors. *Drug discovery today*, 7:235–246, 2002.

[49] H. Zhang, K. Huang, Z. Li, L. Banerjei, K. E. Fisher, N. V. Grishin, E. Eisenstein, and O. Herzberg. Crystal structure of ybak protein from haemophilus influenzae at 1.8A resolution: structure-functional implications. *Proteins: Structure, Function and Genetics*, 40:86–97, 2000.