Dottorato di Ricerca in Informatica
Università di Bologna e Padova

Ciclo XXII
Settore scientifico-disciplinare di afferenza: INF/01
Indirizzo: Bioinformatica

Machine learning methods for prediction of disulphide bonding states of cysteine residues in proteins

Priyank Shukla

March 2010

Coordinatore:	Relatore:	Tutore:
Prof. Simone Martini	Prof.ssa. Rita Casadio	Prof. Luciano Margara
_____	_____	_____

Dedicated to my grandfather, Shri. Ramesh Chandra Shukla

Abstract

The goal of this thesis work is to develop a computational method based on machine learning techniques for predicting disulfide-bonding states of cysteine residues in proteins, which is a sub-problem of a bigger and yet unsolved problem of protein structure prediction. Improvement in the prediction of disulfide bonding states of cysteine residues will help in putting a constraint in the three dimensional (3D) space of the respective protein structure, and thus will eventually help in the prediction of 3D structure of proteins. Results of this work will have direct implications in site-directed mutational studies of proteins, proteins engineering and the problem of protein folding.

We have used a combination of Artificial Neural Network (ANN) and Hidden Markov Model (HMM), the so-called Hidden Neural Network (HNN) as a machine learning technique to develop our prediction method. By using different global and local features of proteins (specifically profiles, parity of cysteine residues, average cysteine conservation, correlated mutation, sub-cellular localization, and signal peptide) as inputs and considering Eukaryotes and Prokaryotes separately we have reached to a remarkable accuracy of 94% on cysteine basis for both Eukaryotic and Prokaryotic datasets, and an accuracy of 90% and 93% on protein basis for Eukaryotic dataset and Prokaryotic dataset respectively. These accuracies are best so far ever reached by any existing prediction methods, and thus our prediction method has outperformed all the previously developed approaches and therefore is more reliable.

Most interesting part of this thesis work is the differences in the prediction performances of Eukaryotes and Prokaryotes at the basic level of input coding when 'profile' information was given as input to our prediction method. And one of the reasons for this we discover is the difference in the amino acid composition of the local environment of bonded and free cysteine residues in Eukaryotes and Prokaryotes. Eukaryotic bonded cysteine examples have a 'symmetric-cysteine-rich' environment, where as Prokaryotic bonded examples lack it.

.

Acknowledgements

I would like to thank my PhD thesis supervisor *Prof. Rita Casadio* for accepting me as a PhD student under her benevolent guidance and allowing me to be a part of her renowned 'Bologna Biocomputing Group'. I am thankful to her for her scientific supervision and support during my three years of stay at Bologna. She always encouraged my ideas and me and pointed them in new directions whenever I faced problem with my research.

I would like to express my gratitude towards my thesis internal committee members *Prof. Luciano Margara* and *Prof. Gianluigi Zavattaro*, for providing invaluable comments and suggestions. I would also like to thank *Prof. Simone Martini* (PhD coordinator), for his wonderful coordination, and, prompt and concerning replies to my email queries.

I would like to thank all the members of 'Bologna Biocomputing Group', in particular *Dr. Pier Luigi Martelli* for patiently guiding me throughout my research, *Dr. Piero Fariselli* for his invaluable suggestions in the predictor development, *Dr. Lisa Bartoili* and *Dr. Marco Vassura* for their help concerning programming issues, and *Dr. Ivan Rossi* and *Dr. Gianluca Tasco* for their hardware and technical support.

I would like to thank my external referees, *Prof. David T. Jones* (University College London) and *Prof. Osman Ugur Sezerman* (Sabanci University) for reviewing this dissertation.

I would like to thank my parents, for their unconditional love and for nurturing my dreams and me. They have made many sacrifices for my education and have always supported me in my endeavours. I am thankful to my younger brother, *Flt. Cdt. Praveen Shukla*, who has been always inspiring me. A special thank to my wife, *Shalinee Tiwari (Shukla*), who has been a pillar of strength during this whole undertaking. Her cheerful attitude drove me away from the anxieties during ups and downs of a PhD student life. I thank her for her unlimited patience and constant support to my heart, brain and stomach.

Finally, I would like to acknowledge financial support received in the form of two scholarships; 'Brain-in' PhD residential scholarship from Institute of Advanced Studies, University of Bologna and '100-Young Indian Researcher' scholarship from MIUR, Government of Italy.

Priyank Shukla

Bologna, March 2010

Contents

List of Figures

List of Tables

Chapter 1: Introduction

Development of sequencing (genome & proteome) projects has revolutionized Bioinformatics, specifically, the field of Structural Bioinformatics, which has an indirect role to play in pharmaceutical industry for designing drugs. Considering proteins to be the main targets of most of the drugs, knowledge of their 3D structures is important. Experimental methods based on X-ray crystallography and NMR-techniques to reveal the protein 3D structures are very expensive and time taking. Considering the huge amount of proteome sequencing data coming up everyday with the different sequencing project running in different laboratories worldwide, importance of computational methods to handle, analyze and interpret this data is growing up. However, prediction of protein 3D structure via computational methods with 100% accuracy has still not been reached. And therefore it still remains an open problem for Structural Bioinformatics world. An alternative solution to tackle this bigger problem is to break it into sub-problems. And in this thesis work we have targeted one of the sub-problems, which is to predict the disulphide bonding states of cysteine residues in a protein sequence.

The introduction and description of this sub-problem follows in this chapter (section 1.3), with a literature survey on existing methods in section 1.4. The description of the machine learning methods used for this thesis work is done in chapter 2, followed by the description of prediction method we have developed in chapter 3. In chapter 4, we discuss the results and show that our method has outperformed all the previously existing approaches. Most interesting part of this thesis work is the differences in the prediction performances of Eukaryotes and Prokaryotes at the basic level of input coding when 'profile' information was given as input to our prediction method. And we discuss this interesting and new finding in section 4.4.

1.1. Protein structure

Proteins are long organic polypeptide of amino acid residues, which are arranged in a linear chain (primary structure) and joined together by **'peptide bonds'** (Fig. 1.1) between the carboxyl and amino groups of adjacent amino acid residues.

This linear chain actually consists of a uniform repetitive backbone, which is also called main chain, with a particular side chain attached to each residue. It is the side chain of the residue, which determines its physiochemical properties.

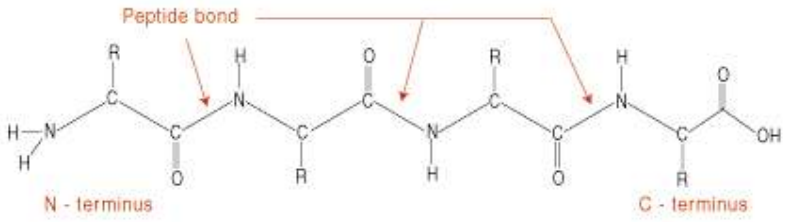


Figure 1.1. Structure of a polypeptide chain showing peptide bonds.

The sequence of amino acids in a protein is defined by the sequence of a gene, which is encoded in the genetic code of the organism. In general, genetic code specifies 20 standard amino acid residues. However, exceptionally certain organism can have extra residues like selenocysteine and pyrrolysine.

Soon after its synthesis, most proteins fold into unique three-dimensional (3D) structures called interchangeably tertiary, folded, or native structure, which in turn is responsible for a specific function of the protein. The tertiary structure of a protein can provide important information about how the protein performs its function. Although many proteins can fold unassisted, simply through the chemical properties of their amino acids, others require the aid of molecular 'chaperones' to fold into their native states.

Proteins can also work together to achieve a particular function, and they often associate to form stable complexes. Biochemically, proteins play a variety of roles in life process:

i) **Structural proteins:** Proteins, whose primary function is to produce the structural components of the cell, e.g. viral coat protein, horny outer layer of human and animal skin, and proteins of the cytoskeleton.

ii) **Enzymes:** Proteins that catalyse bio-chemical reaction, e.g. Insulin.

iii) **Transport and storage proteins:** E.g. Haemoglobin.

iv) **Regulatory proteins:** These include hormones and receptor/signal-transduction proteins.

v) Gene transcription controlling proteins.

vi) Proteins are also involved in recognition, including cell adhesion molecules, and antibodies and other proteins of the immune system.

Proteins are macromolecules, and in many cases only a small part known as **'Active Site'** of the protein is directly functional. Rest of the part exists only to create and fix the spatial relationship among the active site residues. Proteins evolve by structural changes produced by mutations in the amino acid sequences and genetic rearrangements that bring together different combinations of structural subunits.

1.1.1. Hierarchical classification of proteins

It was the Danish protein chemist K.U. Linderstrom-Lang who classified the proteins into basically 3 classes:

i) **Primary structure:** The amino acid sequence, defined by set of primary chemical bonds.

ii) **Secondary structure:** The assignment of helices and sheets on the basis of hydrogen-bonding pattern of the mainchain.

iii) **Tertiary structure:** The assembly and interaction of the secondary structures is called tertiary structure. Tertiary structure is generally stabilized by non-local interactions, most commonly the formation of a hydrophobic core, but also through salt bridges, hydrogen bonds, disulfide bonds, and even post-translational modifications. The term "tertiary structure" is often used as synonymous with the term *fold*. The Tertiary structure is what controls the basic function of the protein.

Later, J.D. Bernal gave a fourth class of protein:

iv) **Quaternary structure:** Proteins that are composed of more than one subunit.

Some additional level of hierarchy:

v) **Supersecondary structures:** Proteins that show recurrent patterns of interaction between helices and strands of sheet close together in the sequence. For e.g. α-helix hairpin, β-hairpin and β-α-β unit.

vi) **Domains:** Many proteins contain compact units with in the folding pattern of a single chain; they look as if they should have independent stability. These are called domains. E.g. RNA-binding protein L1.

vii) **Modular proteins:** These are multidomain proteins, which often contain many copies of closely related domains.

Domains recur in many proteins in different structural contexts; that is, different modular proteins can 'mix and match' sets of domains. E.g. firbonectin, a large extracellular protein involved in cell adhesion and migration, containing 29 domains including multiple tandem repeats of three types of domains called F1, F2 and F3.

1.1.2. Structural classification of proteins

Structural classification of proteins is based on secondary and tertiary structures of proteins.

i) **α-helical:** where the protein's secondary structure is exclusively or almost exclusively α-helical.

α -helix is a right- or left-handed coiled conformation, resembling a spring, in which every backbone N-H (amino) group donates a hydrogen bond to the backbone C=O (carboxyl) group of the amino acid four residues earlier. This secondary structure is also sometimes called a classic **Pauling-Corey-Branson** alpha helix.

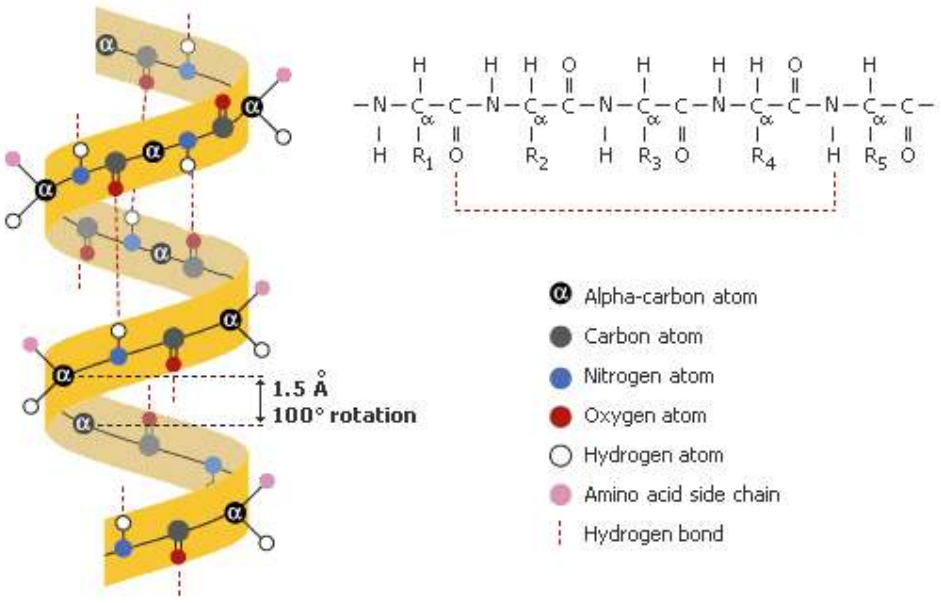


Figure 1.2. An α-helix. Hydrogen bonds are formed between C=O (Carboxyl) group of a residue with the N–H (Amino) group of 4th successive residue.

ii) **β-sheet:** where the protein's secondary structure is exclusively or almost exclusively β-sheet.

The **β sheet** (also **β-pleated sheet**) is the second form of regular secondary structure in proteins consisting of **beta strands** connected laterally by five or more hydrogen bonds, forming a generally twisted, pleated sheet. A beta strand is a stretch of amino acids typically 5–10 amino acids long whose peptide backbones are almost fully extended.

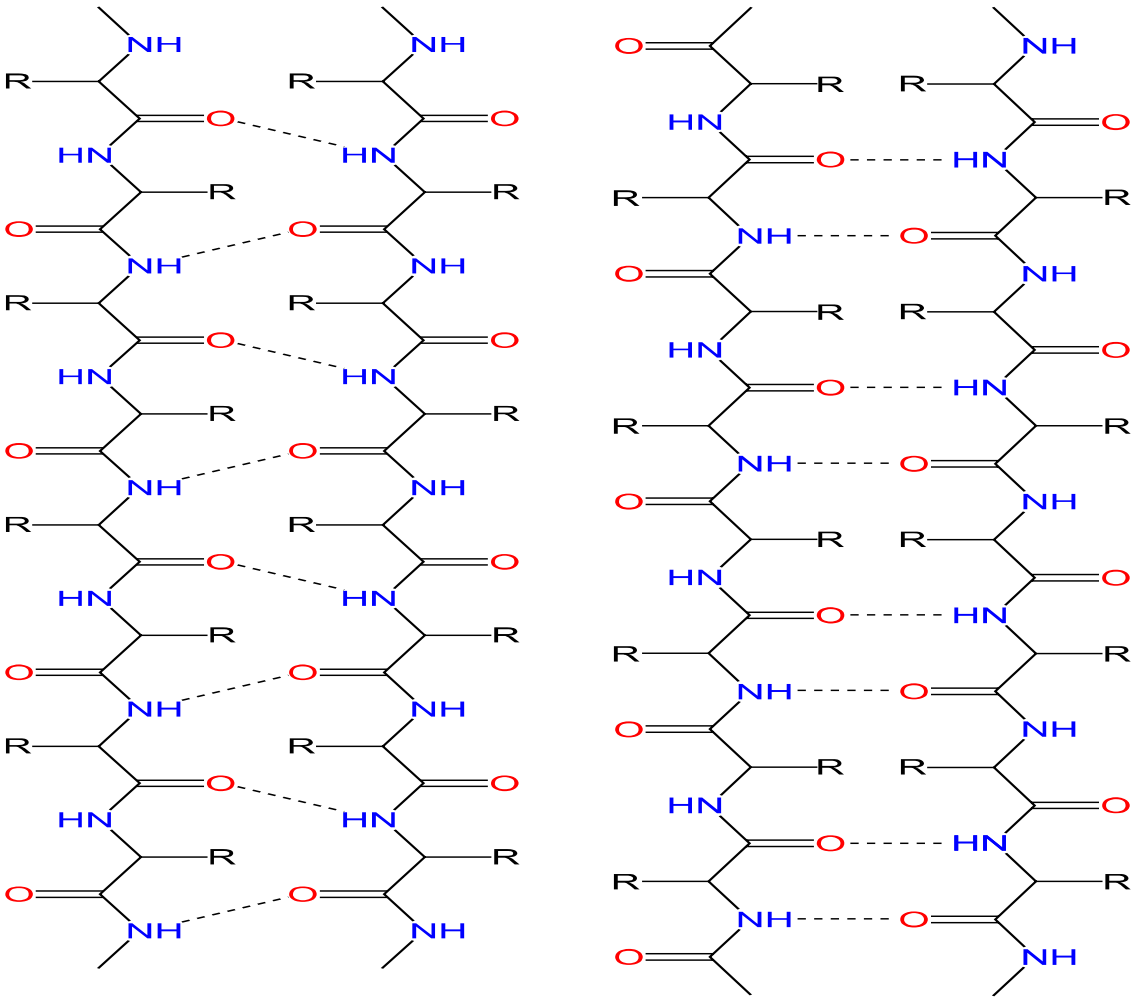


Figure 1.3. β-sheet structures. (a) Parallel β-sheet. All strands point in same direction. (b) Anti parallel β-sheet. All pairs of adjacent strands point in opposite direction. Hydrogen bonds are shown by dotted lines.

iii) **α + β:** Those structures in which α-helices and β-sheets are separated in different parts of the molecule and there is absence of β – α – β supersecondary structure.

iv) **α /β:** Those structures in which α-helices and β-sheets are assembled to form β – α – β supersecondary structure.

v) **Small Proteins:** Proteins with little or no secondary structures.

1.2. Protein structure prediction

Protein structure prediction refers to the problem of determining the three-dimensional (3D) structure of proteins from one-dimensional (1D) chain of amino acid residues (primary structure).

Protein structure prediction is of high importance in the field of medicine (for e.g. Drug designing) and in Biotechnology (for e.g. design of novel enzymes). Moreover, it is the structure of protein, which determines its biological function, and therefore study of protein structure is very important in order to understand the biological functions of proteins.

The classical methods for protein's 3D structure determination involve wet-lab experiments such as 'X-ray crystallography' and 'Nuclear Magnetic Resonance' (NMR). X-ray crystallography involves diffraction of x-rays from crystallised protein, which produces a 3D picture of the atoms involved in the protein crystal. NMR is based on quantum mechanical magnetic properties of an atom's nucleus and is performed on the aqueous samples of highly purified proteins. The drawbacks are; both the above methods can only be applied to soluble proteins, they are expensive techniques and can take a long time (sometimes more than a year). On the other hand, the sequencing of proteins (determination of 1D structure of proteins) is relatively fast, simple, and inexpensive. As a result, there is a large gap between the number of known protein sequences and the number of known 3D protein structures. This gap has grown over the past decade and is expected to keep growing as a result of the various genome projects (study of complete genetic material of organisms) worldwide. Thus, computational methods that may give some indication of structure and/or function of proteins are becoming increasingly important. Since it was discovered that proteins are capable of folding into their unique native state without any additional genetic mechanisms, over 25 years of effort has been expended on the determination of the three-dimensional structure from the sequence alone, without further experimental data. Despite the amount of effort, the protein folding problem remains largely unsolved and is therefore one of the most fundamental unsolved problems in Structural Bioinformatics today.

Some of the general computational methods of protein structure predictions from amino acid sequences are:

i) **Secondary structure prediction:** Methods, in which we try to predict the regions of the sequence forming α-helix and β-sheets, irrespective of what is their arrangement in 3D space. Eg: PSIPRED

ii) **Homology Modelling:** It is the method by which we predict three-dimensional structure of a protein with the help of already known structures of one or more related proteins. It is based on the reasonable assumption that two homologous proteins will share very similar structure. If the sequences of two related proteins have 50% or more identical residues in an optimal alignment, the structures are likely to have similar conformations over >90% of the model [Lesk AM, 2008]. Eg: MODELLER, SwissModel, BioSerf.

iii) **Fold recognition:** Determination of folding pattern of the query sequence of unknown structure by comparing it with a library of known structures. Eg: GenTHREADER

iv) **Ab initio:** Prediction of 3D structure of proteins based on basic physical principles of residues rather than previously solved structures. It is also known as ***'de novo'*** protein modelling methods. Eg: Rosetta.

So far, several approaches based on the general computational methods have been addressed to tackle this central problem of protein structure prediction, but till now without a general solution. However, we see that nature solves this problem everyday for billions of molecules in a single organism. The "ab initio" solution of the protein-folding problem is still lacking. And a typical alternative approach for solving this problem can be to identify a set of sub-problems, such as the prediction of protein secondary structures, solvent accessibility and/or prediction of residue contacts and/or design of heuristic solutions [Vassura et al., 2008].

In this thesis we will focus on a sub-problem, prediction of disulfide bonding states of cysteine residues in proteins, and the description of this sub-problem follows in the next section.

1.3. Prediction of disulphide bonding states of cysteine residues (the sub-problem)

Disulphide bonds also called 'Disulphide Bridges' or 'SS-bonds' are covalent bonds formed between two cysteine residues of a protein.

Chemically, the bonding takes place between the thiol (SH) groups of two cysteine residues (Fig. 1.4).

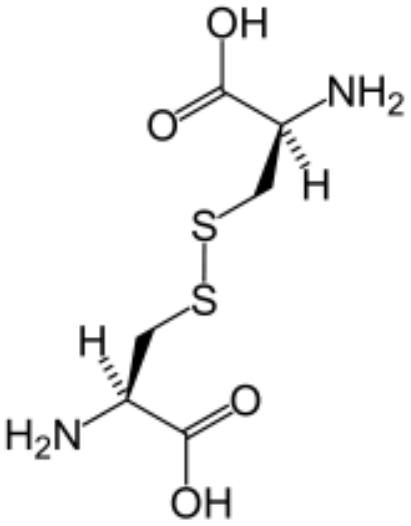


Figure 1.4. Disulphide bond between two Cysteine residues

Disulphide bonds are classified into two types:

i) **Intra-bonded:** when the bonding takes places between two-cysteine residue of the same protein chain.

ii) **Inter-bonded:** when the bonding takes place between two cysteine residues of two different chains of the same protein.

Disulfide bonds play an important role in the folding and stability of some proteins, usually proteins secreted to the extracellular medium [Sevier et al., 2002]. Since most cellular compartments have reducing environments, disulfide bonds are generally unstable in the cytosol with some exceptions. Moreover, reduction of these disulphide bridges triggers functionally relevant conformational changes in the protein structure [Creighton T, 1996], and thus help in defining the function of protein.

Disulphide bond stabilize protein native structures by lowering global free energy. And they may be involved in the protein-folding pathways. In some proteins, the oxidation and reduction of cysteine residues are the essential part of their catalytic functionalities. The contribution of the disulfide bridge to the thermodynamic stability of proteins has been described as being due to a reduction in the conformational entropy of the unfolded polypeptide chain causing a destabilization of the unfolded state relative to the native state [Betz S.F., 1993]. It has been estimated both experimentally [Privalov P.L., 1988] and theoretically [Casadio et al, 1995]. Several analyses of the characteristics of disulfide bonds in proteins have been performed, including structural and sequence features and classification of connectivity [Harrison et al., 1994]. This strengthens the view that disulfide bonds increase the conformational stability of the protein mainly by constraining the unfolded conformation, as many experimental and theoretical studies suggest [Wedemeyer et al., 2000]. Disulfide bond has been recognized as a major contributor to protein thermal stability [Vieille et al., 2001]. A genome-wide survey has shown that hyperthermophile proteins tends to contain more disulfide bonds than proteins from thermophilic and mesophilic organisms [Beeby et al., 2005].

Considering the importance of disulphide bonds in defining the structure and function of the protein, prediction of disulphide bonding state of cysteine can be an important step in the field of protein structure prediction, as it will help in putting a constraint in three-dimensional space of protein structure. In the next section we have done a literature survey on the computational methods so far developed in the area of predicting disulphide-bonding state of cysteines.

1.4. Literature review (existing approaches)

Disulphide bond prediction has been extensively studied in recent years and a number of successful machine-learning approaches exist.

It was early 90s when statistical [Fiser et al., 1992] and neural network based [Muskal et al., 1990] methods were proposed to predict the cysteine bonding state with flanking residues.

However, major breakthrough in this field came up in the year 1999, when Fariselli et al. described the role of evolutionary information for training a neural-network based predictor for predicting disulphide bonding states of proteins. Standard feed forward networks were implemented with a back-propagation algorithm as learning procedure. The network architecture consisted of a perceptron with two output nodes, which discriminate bonded and free cysteine propensities, respectively, with no hidden layers. Six different types of input coding based on single-sequence input or multiple-sequence profiles and local features such as 'Residue Charge', 'Hydrophobicity', 'Conservation Weight', and 'Relative Entropy' were considered. Training was performed using 2452 cysteine residues containing segments extracted from 641 non-homologous proteins of well-resolved three-dimensional structures. Using protein single sequence, the prediction accuracy on cysteine basis was 72%, which improved drastically up to 78% by incorporation of evolutionary information. Furthermore, an improvement of 2% was obtained when the conservation weight and relative entropy were also used. Finally a jury method improved the prediction accuracy up to 81%.

In the year 2000, Fisher and Simon suggested that cysteines tend to occur in the same oxidation state within the same protein. Based on this "all or none" rule they proposed a new method [Fisher et al., 2000] for predicting the oxidation state of cysteine residues by conservation scores derived from multiple sequence alignments. A database of 81 protein alignments was used in their analysis. Conservation scores based upon the physico-chemical properties of amino acids were calculated for each position in each alignment. For each position in each protein, this score was then divided by the average conservation of the protein to give a relative conservation score C_r. The efficiency of the prediction was tested by the *jack-knife* procedure, and the prediction accuracy of the redox state of cysteines was above 82%. Since, the

prediction of this method is either all oxidized or all reduced, the practical usage in protein engineering is relatively limited.

In the year 2002, Mucchielli-Giorgi et al., investigated the relative contribution of the segments flanking the target cysteines and the over all amino acid residue composition of the protein. A simple prediction method [Mucchielli-Giorgi et al., 2002] based on the training of logistic function was used, in which a decision rule was simply comparing the output of the logistic function with a given threshold. If the output value comes out to be great than or equal to the given threshold, the cysteine of interest was predicted as bonded, else free. A dataset of 559 proteins was used, and the evaluation was done with a 5 fold cross-validation procedure. They got a significant higher accuracy of 83% when considering a set of global features (overall residue composition, normalized protein size and cysteine occurrence), than an accuracy of 70% and 68% for local features of residues ("frequency description" and "binary description" respectively). These results suggested that, for the disulphide bonding state, the information on the residues flanking the cysteines (local information) is less informative than the amino acid content of the whole protein (global information). Additionally, they also used a combination of logistic function learned with a subset of proteins homogenous in terms of their residue contents, and reached to a prediction accuracy of 84%.

Methods described so far were not able to capture global information with respect to cysteine examples belonging to the same protein, since they were predicting one cysteine at a time without keeping record of different cysteine predictions associated with same protein sequence. In other words, when a cysteine is predicted in a protein chain, no information about the predicted bonding state of other cysteines of the same chain was considered. And since to make a disulphide-bond, two cysteines are required, disulphide bonding cysteine chains should have even number of cysteines. In the year 2002, Martelli et al., used this trivial "even bonded cysteines" feature along with other global and local characteristic of protein chains, and proposed a method [Martelli et al., 2002] by implementing a hybrid system (hidden neural network) that combines a neural network (NN) and hidden markov model (HMM). On a sliding window of 27 residues centered at cysteine, local information in the form of profiles containing evolutionary information generated by multiple sequence alignment was given to a feed-forward NN. The output generated by NN was used as emission probabilities for a four-state HMM, which incorporated an

"even bonded cysteine" constraint as a global feature. A training set of 4136 cysteine-containing segments extracted from 969 nonhomologous proteins of well resolved structure was used. Testing was done on two different sets; whole dataset (WD), which included proteins with single cysteine and a difficult reduced dataset (RD), which excluded proteins with single cysteine. A 20-fold cross validation was done in order to validate the results. Initially, when using only NN-based predictor average accuracies of 80% on cysteine basis and 57% on protein basis were obtained, which were similar to the previously obtained results of Fariselli et. al., in 1999. However, a remarkable increase in the accuracies was achieved on incorporating the hybrid system (HNN). Average accuracies reached up to 88% and 84%, on cysteine basis and on protein basis, respectively. This improvement was seemingly caused by the introduction of the global constraint by the regular grammar implemented in the HMM, which not only captures the number of cysteines in a protein chain but also keep track of the bonding states of all the cysteine in the chains.

In the year 2004, Song et al. introduced a two-class predictor, which explores the dipeptide composition of protein sequence, for predicting the oxidation state of cysteines in proteins by means of a linear discriminator [Song et al., 2004]. The idea of this new global feature of dipeptide composition came from the fact that none of the previous methods were using information contained in the order of residues in chains. The dataset they used for training consist of 8114 cysteine-containing segments extracted from 1856 non-homologous proteins. They used the jack-knife procedure to validate their results, and achieved an accuracy as high as 89.1% on cysteine basis and 85.2% on protein basis.

In the same year of 2004, Chen et al. proposed a method based in Support Vector Machine (SVM) and achieved an extraordinary accuracy of 90% on protein basis. Their approach [Chen et al., 2004] consisted of two stages. In the first stage, SVM was used to predict the bonding state of cysteines. In addition to the local residue information defined by flanking residues of the interested cysteines, the amino acid composition was also used as input feature. The decision value obtained from SVM was further normalized by arctan transfer function and used as the state probability of each cysteine state in the next stage. In the second stage, a constraint of even number of cysteine was applied and with the help of branch-and-bound algorithm an optimized value of cysteine state sequences was computed. Same data set as used by Martelli et al. in 2004 was used for evaluation of the method.

In the most recent work of 2006 done by Ceroni et al., they have used an SVM binary classifier to predict the bonding state of cysteine residue, followed by a refinement stage that classifies all the cysteines in a chain by deciding the overall bonding state assignment of an entire chain rather than making several independent prediction. They have used both local feature (profile information for a local window of sequence centered at cysteine of interest) and global features (amino acid composition, chain length, number of cysteines and average cysteine conservation), but reach to an average accuracy of 88% on cysteine basis.

1.5. Overview of the thesis

This whole thesis work is organized into four chapters. Chapter 1 comprises the introduction of the protein structures, the problem of protein structure prediction and existing methods of their prediction, followed by a sub-problem of prediction of disulphide bonding states of cysteine residues in protein structures (which is the ultimate goal of this thesis work). In section 1.3, we have defined and described this sub-problem, followed by a literature survey of the already existing methods (section 1.4).

In chapter 2, we have described machine learning methods, specifically 'Artificial Neural Networks (ANN)', 'Hidden Markov Models (HMM)' and 'Hidden Neural Networks (HNN)', and the algorithms associated with them, which we have used to develop the prediction methods for our sub-problem.

Chapter 3 comprises the description of the prediction methods we have developed for predicting the disulphide bonding states of cysteine residues. Section 3.1 starts with the description of the dataset used for training and testing the machine learning methods, followed by the description of the importance of cross-validation so as to validate the results obtained on the datasets used (section 3.2). The description of various features used as input for the methods follows in section 3.3. In Section 3.4, we describe the implementation of the methods. Section 3.5, describes the statistical indexes we have used to measure the performances of our methods.

Chapter 4 consists of results and discussion. In section 4.1 and 4.2, we describe the performances of the methods developed in this thesis work on Eukaryotic

dataset and Prokaryotic dataset respectively, followed by a comparison of final results with the previously developed approaches (section 4.3), and we show that our final prediction method outperforms all the previously developed approaches. In section 4.4, we explain why Eukaryotic dataset performed better than Prokaryotic dataset at the basic level of input coding when ‘profile’ information was given as input to our prediction methods, which is one of the most interesting part of this thesis work. Finally in section 4.5, we conclude the thesis with some remarks and future aspects of this work.

Chapter 2: Machine Learning Methods

The concept of Machine Learning (ML) is a broad sub-field of Artificial Intelligence. It is concerned with design and development of algorithms and techniques that allow computers to learn “rules” from an existing dataset and to use them in order to “classify” a new unknown dataset.

The major focus of machine learning research is to learn complex pattern from a given data and extract information from it automatically, by computational and statistical methods, and further use it in order to build a classifier, which can classify an unlabelled dataset.

Regarding interference of human intuition in the machine learning process there are two understanding. Some machine learning systems attempt to eliminate the need for human intuition in data analysis, while others try to adopt a collaborative approach between human and machine. However, human intuition cannot be entirely eliminated, since the system's designer must specify how the data is to be represented and what mechanisms will be used to search for a characterization of the data. Machine learning can be viewed as an attempt to automate parts of the scientific method.

Machine learning methods can be broadly classified into two; supervised and unsupervised, based on the amount of human intervention. The contrasting feature between the two is; in supervised learning the data comes with class label and we learn how to associate this labelled data with classes, where as in unsupervised learning all the data is unlabelled and the learning procedure consists of both defining the labels and associating objects with them [Tarca et al., 2007]. In section 2.1 and 2.2, a further discussion follows on supervised learning and unsupervised learning, respectively.

Both supervised and/or unsupervised learning methods have equal importance in life science research and have huge amount of literature available. For e.g., protein secondary structure prediction by using amino acid sequence information [Rost et al., 2004], and classifying patients into different clinical groups and to identify new disease groups by using gene expression data [Perou et al., 1999 and Alizadeh et al., 2000].

2.1. Supervised Learning

This technique is based on deducing a function from training dataset that maps input to a desired output. *A priori*, the training dataset consist of pairs of input object (typically vectors) and desired outputs. In a simpler way we can say that, objects in a given dataset are classified using a set of attributes or features as input, and the result of the classification process is a set of rules that prescribe assignments of objects to classes as output, based solely on values of features. In a biological context, example of object-to-class mapping can be protein sequences to their secondary structures, and features (input) can be presence or absence of a particular amino acid at a particular position in the protein sequence.

The goal in supervised learning is to design a system able to accurately predict the class membership of new objects based on the available features.

Understanding the above concept of supervised learning in mathematical notations, we can consider a dataset of n objects (i = 1, … n) that are classified *a priori* into K *(y = 1, ... K)* classes. For instance, if we want to distinguish between different types of secondary structure of proteins based on protein sequence information, then K would represent the number of known existing type of protein secondary structures (for e.g. α-helix, β-sheets, α+β, α/β etc). Suppose we have p (j = 1, … p) number of features to describe each object i of our dataset, then we can organize our whole dataset of n objects on the basis of p features in an $n * p$ matrix $X = (\boldsymbol{x_{ij}})$, where $\boldsymbol{x_{ij}}$ represents the measured value of the variable (feature) j in the object (sample) i. Every row of the matrix X is therefore a vector $\boldsymbol{x_i}$ with p features to which a class label y_i is associated. For such multiclass classification problems, a classifier can be seen as an ensemble of K discriminant functions $g_k(\boldsymbol{x})$ such that the object i, described by the feature vector $\boldsymbol{x_i}$, will be assigned to the class K for which the function $g_k(\boldsymbol{x})$ is maximised**.** The classifier thus divides the feature space X into K subsets.

For the identification of the discriminant function $g_k(\boldsymbol{x})$ there are two main approaches. The first is to compute the probability density function of **x** for a given class and assign $g_k(\boldsymbol{x}) = f(p(\boldsymbol{x} \mid y = k))$, where f is a monotonic increasing function (for e.g. logarithmic function). Intuitively, the resulting classifier will classify an object **x** in the class in which it has highest membership probability. In practise, $p(\boldsymbol{x} \mid y = k)$ is

unknown, and therefore needs to be estimated from a set of correctly classified samples named as “training set”. Parametric methods (such as, linear and quadratic discriminant) and nonparametric methods (such as k-nearest neighbour decision rule) are used for density estimation of the above function. The second approach is to use data to estimate the class boundaries directly, without explicit calculation of the probability density functions. Examples of algorithms in this category include decision trees, neural networks, and support vector machines (SVM).

2.2. Unsupervised Learning

Unsupervised learning is synonymously also known as “Clustering problem”. Since, the data in this case do not contains any predefined labelled class, the task is to group the given data into clusters based on the common features they share. Principally, one needs to explore the data and discover similarities between the objects. Therefore, the key point of clustering procedures is the definition of the degree of similarity between the analysed objects. And to measure this similarity, basically ‘Euclidean distance’ or ‘one minus correlation’ is used. However, to define a degree of similarity is a subject of consideration.

Clustering procedure can be broadly classified into ‘Hierarchical Clustering’ and ‘Partition Clustering’; former divides the data into a hierarchical tree-like structure and later into certain number of clusters, respectively.

Hierarchical clustering can be further divided into ‘bottom-up’ and ‘top-down’, based on type of approach used to divide the dataset. In ‘bottom-up’ approach, each data point (or object) is initially considered as a cluster in itself. Subsequently, the clusters are iteratively grouped based on their similarity. In contrast, the top-down approach starts with a unique cluster containing all the objects, which is iteratively divided into smaller clusters until each cluster contains a single object.

Partition clustering, starts with a predefined number of clusters as specified by the user. The most used algorithm is *K-means* clustering, where *K* is the number of clusters predefined by the user. Objects are assigned to these clusters based on their similarity (Euclidean distance) from each cluster. Subsequently, a two-step iterative procedure works as follows:

i) Recalculating the position (group mean) of clusters based on the current membership of each cluster.

ii) Reassigning the objects to the *K* clusters.

The algorithm ends, when no further change in the assignment of objects is possible.

For an extensive review on supervised and unsupervised learning methods, please refer to Tarca et al., (2007).

In coming sections 2.3, 2.4, and 2.5 we will discuss 'Artificial Neural Networks' (ANN), 'Hidden Markov Models' (HMM) and Hidden Neural Networks (HNN); three supervised learning techniques we have used in this thesis work.

2.3. Artificial Neural Networks (ANN)

Artificial Neural Networks (ANN), also called as Neural Networks (NN) are mathematical/computational models, which belong to class of general computational structures based loosely on the anatomy and physiology of biological nervous system.

The biological nervous system, which broadly consists of brain and spinal cord, is managed by a group of specialized cells called 'Neurons' or 'Nerve Cells'. Work of nerve cells is to communicate information about an organisms surroundings and itself by conducting and generating impulses between each other. Even though, in the beginning biological nervous system was the useful source of inspiration for the development of ANN, but it is clear today that artificial neurons used in most of ANNs are quite remote from biological neurons [Bower JM & Beeman D, 1995]. ANN has become an important tool in the arsenal of machine learning and can be applied to various fields to a wide variety of classification and pattern recognition problems, including computational biology.

At the most basic level, ANNs can be viewed as a broad class of parameterised graphical models consisting of networks with interconnected units (artificial neurons) evolving in time. A single artificial neuron (Fig. 2.1), in the computational scheme, is a node in a directed graph, with one or more entering connections designated as input, and a single leaving connection called as output.

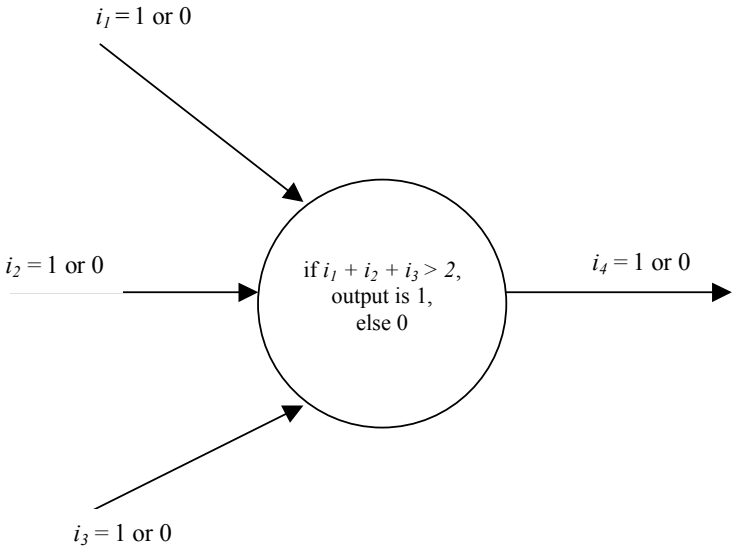


Figure 2.1. A single artificial neuron, which takes 3 inputs (i_1, i_2, i_3) in binary form (0 or 1) and gives a binary output based on the conditional formula. (if $i_1 + i_2 + i_3 > 2$ output is 1, else 0).

In the physiological metaphor, a neuron is said to be 'fired' if the output is 1, else it 'didn't fired' if the output is 0. A simulated neuron differs in the number of input and output connection and the formula for deciding whether to fire or not.

To make an ANN, several neurons are assembled together, so that output of some works as an input for others. Architecture of ANN can be a *feed-forward*, if it is devoid of directed loops, and it can be *layered*, if the units are partitioned into classes (also called as layers) and connectivity patterns are defined between the classes.

For most of the problems related to molecular biology, layered feed-forward architectures of ANN (Fig. 2.2) are used. Also for this thesis work we have used the same. The ANN units are often partitioned into *'visible'* units and *'hidden'* units. The visible units are those, which are in direct contact with the external world, such as input and output units, and hidden units are those, which do not interact directly with the outside world. Most of the time, in simple architecture the input and output units are grouped in layers called *'input layer'* and *'output layer'* respectively. A layer containing only the hidden units is called the *'hidden layer'*.

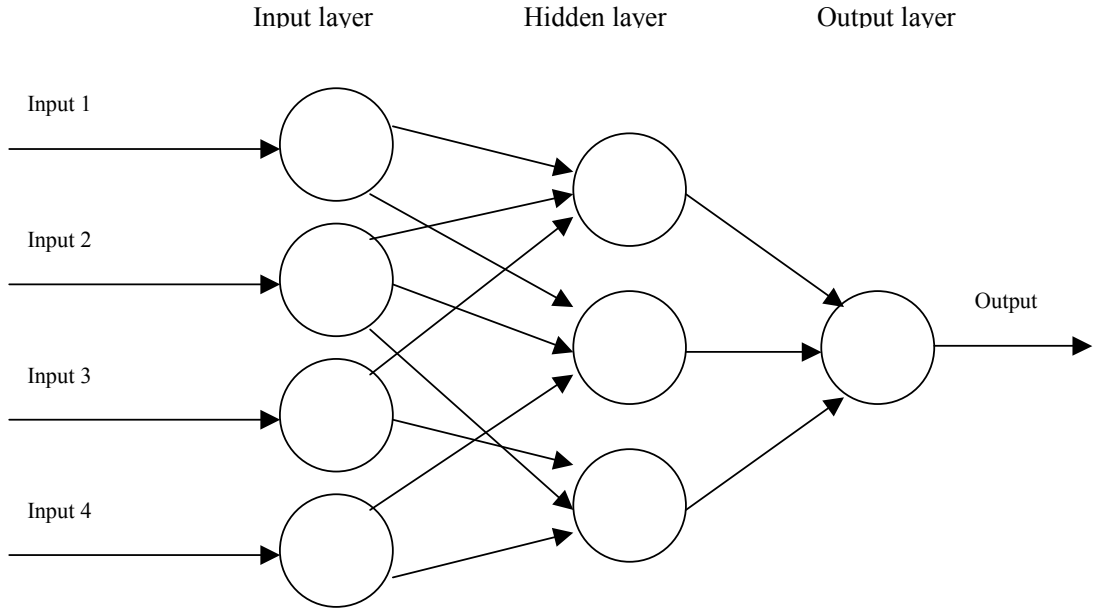


Figure 2.2. A layered feed-forward Artificial Neural Network (ANN).

An unlimited degree of complexity can be created depending on how the assembling and connection of neurons is done in an ANN, and by varying the strength of connections between the neurons. For a very simple example, considering the same (Fig. 2.1) of an input of 3 neurons (i_1, i_2 and i_3) which together generate an output, by the formula $i_1 + i_2 + i_3$, is equally sensitive for all the 3 inputs. The same architecture can be made more sensitive for some nodes by adding weights, for instance $10i_1 + 5i_2 + i_3$. In this case input node i_1 becomes most sensitive and input node i_3 least sensitive. Biologically, this may correspond to changing strengths of synapses between nerve cells.

Mathematically, neural network models can be defined by a function,

$$f : X \rightarrow Y$$

So as to say that, each unit i of ANN, receives a total input x_i from the units connected to it, and then produces an output $y_i = f_i(x_i)$, where f_i is a transfer function.

In general, all the units in the same layer have the same transfer function, and the total input is a weighted sum of incoming outputs from the previous layer, so that

$$x_i = \sum_{j \in N-(i)} w_{ij} y_j + w_i$$

Therefore,

$$y_i = f_i(x_i) = f_i \left(\sum_{j \in N-(i)} w_{ij} y_j + w_i \right)$$

Where, w_i is called the bias, or threshold, of the unit. Weights w_{ij} and w_i are the parameters of ANN.

While a neural network does not have to be adaptive per se, its practical use comes with algorithms designed to alter the strength (weights) of the connections in the network to produce a desired signal flow. In a layered feed-forward architecture, all the units in a layer are updated simultaneously, and layers are updated sequentially in the obvious order.

2.3.1 Backpropagation Algorithm

The most important and interesting aspect of ANN is its possibility of learning by itself. Given a specific task to solve, and a class of functions F, learning means using a set of observations to find $f^* \in F$ which solves the task in some optimal sense.

Backpropagation algorithm is the most common method of teaching ANN. Arthur E. Bryson and Yu-Chi Ho first described it in 1969, but it gained recognition in 1986 by the work of David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams, and it led to a "renaissance" in the field of artificial neural network research [Russell S and Norvig P, 2003]. It has been one of the most studied and used algorithm for neural network learning ever since.

It is a supervised learning method, and is most useful for feed-forward networks. It requires a differentiable activation function to be used by the artificial neurons/nodes/units. The backpropagation algorithm looks for the minimum of the error function in weight space using the method of gradient descent. The combination of weights, which minimizes the error function, is considered to be a solution of the learning problem.

Summary of the Backpropagation technique:

i) Give a training dataset to the ANN.

ii) Compare the network's output to the desired/target output from that dataset, and calculate the error in each output neuron/unit.

iii) For each neuron, calculate what the output should have been, and a *scaling factor*, how much lower or higher the output must be adjusted to match the desired output. This is called the local error.

iv) Adjust the weights of each neuron to lower the local error.

v) Assign "blame" for the local error to neurons at the previous level, giving greater responsibility to neurons connected by stronger weights.

vi) Repeat from step 3 on the neurons at the previous level, using each one's "blame" as its error.

Actual algorithm for a 3-layer network (only one hidden layer):

Initialise the weights in the network (often randomly)
Do
    For each example e in the training set
        O = neural-net-output (network, e); forward pass
        T = teacher output for e
        Calculate error (T - O) at the output units
        Compute delta_wh for all weights from hidden layer to output layer;
        backward pass
        Compute delta_wi for all weights from input layer to hidden layer;
        backward pass continued
        Update the weights in the network
Until all examples classified correctly or stopping criterion satisfied
Return the network

As the algorithm name implies, errors and therefore the learning, propagates backward from the output nodes to the inner nodes. Technically, Backpropagation algorithm is used to calculate the gradient of the error of the network with respect to the network's modifiable weights. This gradient is almost always then used in a simple stochastic gradient descent algorithm to find weights that minimize the error. Often the term "Backpropagation" is used in a more general sense, to refer to the entire procedure encompassing both the calculation of the gradient and its use in stochastic gradient descent.

A detailed description of the ANN architecture used in this thesis work follows in chapter 3 section 3.4.1.

2.4. Hidden Markov Models (HMM)

A Hidden Markov Model (HMM) is a statistical model in which the system being modelled is assumed to be a Markov process, i.e. one for which the likelihood of a given future state, at any given moment, depends only on its present state, and not on any past states. This important feature of HMM is also known as 'Markov property'.

The simplest Markov process is a first order process, where the choice of state is made purely on the basis of the previous state. For a first order process with M states, there are M^2 transitions between states since it is possible for any one state to follow another. Associated with each transition is a probability called the **state transition probability** - this is the probability of moving from one state to another. These M^2 probabilities may be collected together in an obvious way into a **state transition matrix.**

Mathematically, a simple regular markov model can be defined by (Π, A), where
$\Pi = (\pi_i)$, is the vector of initial state probability,
$A = (a_{ij})$, is the state transition matrix, $Pr (x_i | x_j)$

Considering a very simple example (Fig. 2.3) of deducing weather conditions (sunny or rainy) between two consecutive days, a state transition matrix can be represented as in Fig. 2.4.

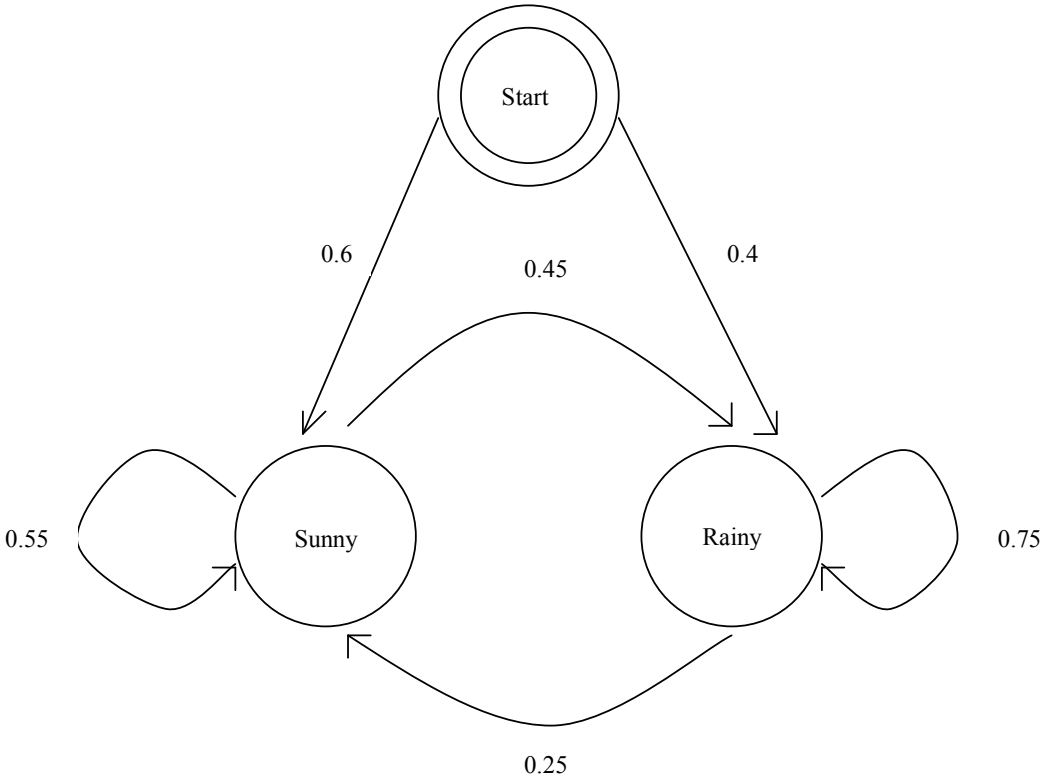


Figure 2.3. A regular markov model. The figure above shows all possible first order transitions between the states of the weather example. Notice, that for the 2 states (Sunny or Rainy), there are $2^2 = 4$ possible transitions.

		Today	
		Sunny	Rainy
Yesterday	Sunny	0.55	0.45
	Rainy	0.25	0.75

Figure 2.4. The state transition matrix above shows possible transition probabilities for the weather example. If it was sunny yesterday, there is a probability of 0.55 that it will be sunny today, and 0.45 that it will be rainy. Notice that (because the numbers are probabilities) the sum of the entries for each row is 1.

In a regular Markov model, the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. In a *hidden* Markov model, the state is not directly visible, but output dependent on the state is visible. Each state has a probability distribution over the possible output tokens. Therefore the sequence of tokens generated by an HMM gives some information about the sequence of states. The adjective 'hidden' refers to the state sequence through which the model passes, not to the parameters of the model. That is to say that even if the model parameters are known exactly, the model is still 'hidden'.

Mathematically, a hidden markov model can be defined by (Π, A, B), where
$\Pi = (\pi_i)$, is the vector of initial state probability, and
$A = (a_{ij})$, is the state transition matrix, Pr ($x_i \mid x_{j\ t-1}$)
$B = (b_{ij})$, is the confusion matrix, Pr ($y_i \mid x_j$)

Each probability in the state transition matrix and in the confusion matrix is time independent, i.e. the matrices do not change in time as the system evolves.

Considering the same weather example we can make it a little complex in order to understand a hidden markov model. Say, we want to deduce the weather condition from a piece of seaweed. A folklore data says that `soggy' seaweed means wet weather, while `dry' seaweed means sunny weather. However, the state of the weather is not restricted to the state of the seaweed, so we may say on the basis of an examination that the weather is probably raining or sunny. A second useful clue would be the state of the weather on the preceding day (or, at least, its probable state). By combining knowledge about what happened yesterday with the observed seaweed state, we might come to a better forecast for today.

In the above example, the observed sequence would be the seaweed and the hidden system would be the actual weather (Fig. 2.5 and 2.6). Therefore, now for this new example what we wish to predict is not what we observe, i.e. to say that the underlying system is hidden.

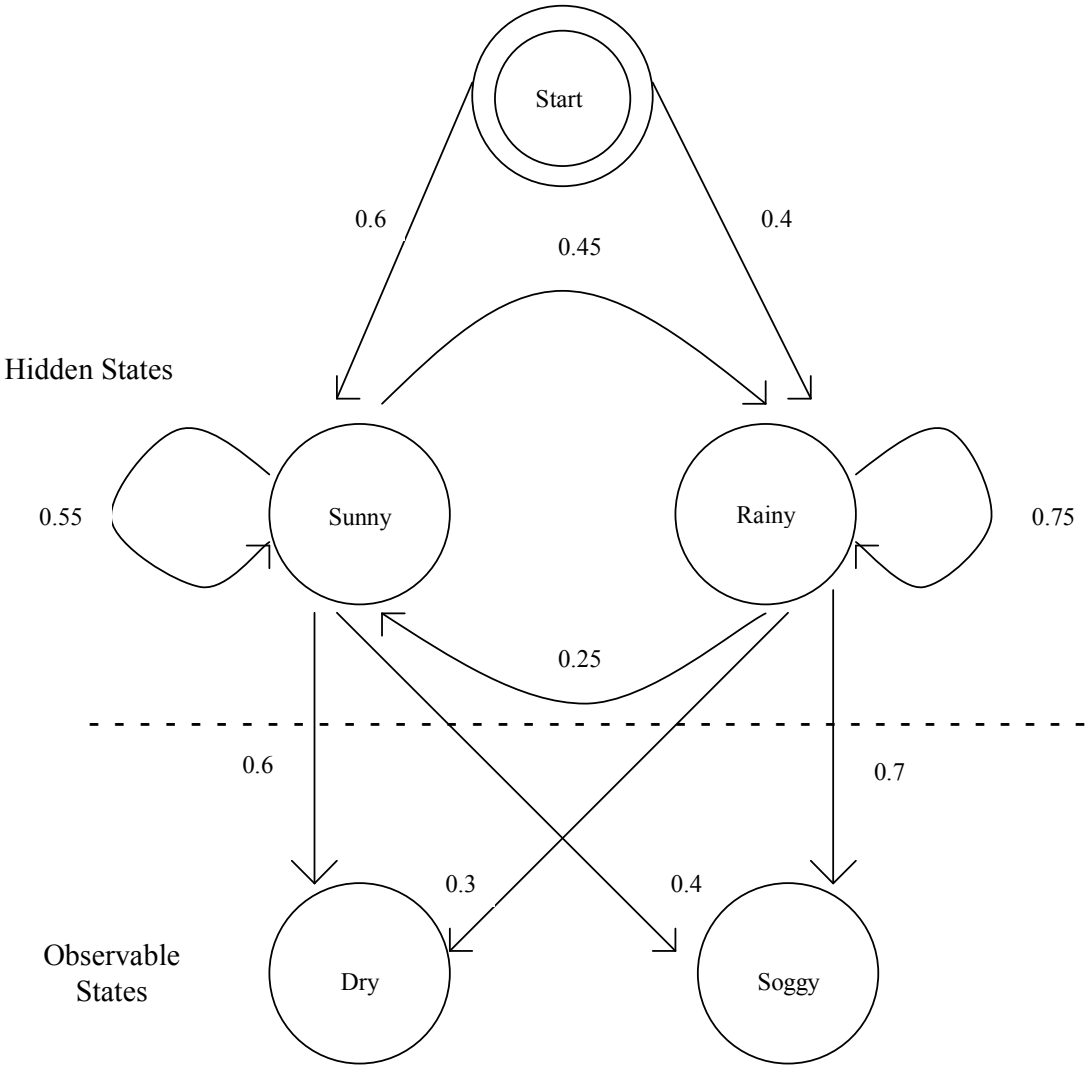


Figure 2.5. A Hidden Markov Model (HMM). The figure above shows the hidden and observable states in the weather example. It is assumed that the hidden states (the true weather) are modeled by a simple first order Markov process, and so they are all connected to each other.

Weather	*Seaweed* Dry	Soggy
Sunny	0.6	0.4
Rainy	0.3	0.7

Figure 2.6. The confusion matrix above shows the probabilities of the observable states given a particular hidden state. Notice that (because the numbers are probabilities) the sum of the entries for each row is 1.

The connections between the hidden states and the observable states represent the probability of generating a particular observed state given that the markov process is in a particular hidden state. Therefore, all probabilities `entering' an observable state will sum to 1, since in the above case it would be the sum of *Pr*(Observable states|Sunny) and *Pr*(Observable states |Cloudy).

Once a system can be described, as HMM, three canonical problems can be associated with it.

i) **Evaluation:** Given the parameters of the model, i.e. an HMM, compute the probability of an observed sequence. This requires summation over all possible state sequences, but can be done efficiently using the forward algorithm, which is a form of dynamic programming.

ii) **Decoding:** Given the parameters of the model and a particular output sequence, find the most probable sequence of hidden states that have generated that output sequence. This requires finding a maximum over all possible state sequences, but can similarly be solved efficiently by the Viterbi algorithm.

iii) **Learning:** The third, and the hardest, problem associated with HMMs is to take a sequence of observations (from a known set), known to represent a set of hidden states, and fit the most probable HMM; that is, determine the (π,A,B) that most probably describes what is seen. No tractable algorithm is known for solving this problem exactly, but a local maximum likelihood can be derived efficiently using the Baum-Welch algorithm or the Baldi-Chauvin algorithm. The Baum-Welch algorithm is also known as the forward-backward algorithm, and is a special case of the Expectation-maximization algorithm.

For this thesis work, the problem was related to both learning using a modified version of expectation-maximization algorithm [Martelli et al., 2002c] and decoding, which requires a Viterbi algorithm to solve. A short description of Viterbi algorithm follows next.

2.4.1 Viterbi Algorithm

The **Viterbi algorithm** is a dynamic programming algorithm for finding the most likely sequence of underlying hidden states in an HMM, that might have generated it. The result of the algorithm is a sequence of observed events called **Viterbi path**.

The name of the algorithm comes from its author Andrew Viterbi who invented it in 1967 [Viterbi AJ, 1967]. The general Viterbi learning idea is to replace calculations involving all possible paths with calculations involving only a small number of likely paths, typically only the most likely one, associated with each sequence.

The algorithm makes a number of assumptions:

First, both the observed events and hidden events must be in a sequence. This sequence often corresponds to time.

Second, these two sequences need to be aligned, and an instance of an observed event needs to correspond to exactly one instance of a hidden event.

Third, computing the most likely hidden sequence up to a certain point t must depend only on the observed event at point t, and the most likely sequence at point $t - 1$.

These assumptions are all satisfied in a first-order hidden Markov model.

In a running example, Viterbi algorithm is used as follows:

```
def example():
        return forward_viterbi(observations,
                               states,
                               start_probability,
                               transition_probability,
                               emission_probability)
print example()
```

A detailed description of the algorithm can be found elsewhere [Forney GD, 1973].

An HMM can be considered as the simplest dynamic Bayesian network. They are especially known for their application in temporal pattern recognition such as speech,

handwriting, gesture recognition, part-of-speech tagging, and Bioinformatics. HMMs have proved to be of great value in analysing real systems. Their usual drawback is the over-simplification associated with the Markov assumption - that a state is dependent only on predecessors, and that this dependence is time independent.

A detailed description of the HMM model used in this thesis work follows in chapter 3 section 3.4.2.

2.5. Hidden Neural Networks (HNN)

A general framework of hybrids of hidden markov models (HMM) and neural networks (NN) are called hidden neural networks (HNN) [Krogh A & Riis SK, 1999].

In HNN, the usual HMM probability parameters are replaced by the outputs of state-specific neural networks. As opposed to many other hybrid networks, the HNN are normalized globally and therefore have a valid probabilistic interpretation. All parameters in the HNN are estimated simultaneously according to the discriminative conditional maximum likelihood criterion. The HNN can be considered as an undirected probabilistic independence network (a graphical model), where the neural networks provide a compact representation of the clique functions.

Although HMM are good in capturing the temporal nature of processes such as speech recognition, they have a very limited capacity of recognizing complex patterns involving more than first-order dependencies on observed data. This is due to the first-order state process and the assumptions of state-conditional independence of observations. Multilayer perceptrons are almost the opposite; they cannot model temporal phenomenon very well but are good in recognizing patterns. Combining the two frameworks in a sensible way can therefore lead to a more powerful model with better classification abilities.

A detailed description of the HNN model used in this thesis work follows in chapter 3 section 3.4.2.

Chapter 3: Prediction Methods

3.1. Dataset

The dataset adopted in this work is a subset of all (71788) protein chains downloaded from Protein Data Bank (PDB) in December 2006. PDB is the single worldwide archive of structural data of biological macromolecules [Berman et al., 2000].

This dataset required filtering so as to remove redundancy, and those chains, which were not fit for prediction. From these chains we removed all the chains having chain breaks by using the annotations given by 'Define Secondary Structure of Proteins' (DSSP) program [Kabsch W & Sander C, 1983]. DSSP is a database of secondary structure assignments for all protein entries in the PDB. We removed these chains having chain breaks, because we were not sure of the folding pattern, and consecutively the disulphide bond (if any) it makes. We also removed those chains, which had no information about 3D coordinates of SG (Sulphur) atom of cysteine residue (if present), which actually participate in disulphide bond formation between two cysteine residues.

With these 2 filtering steps we ended up with 42041 chains. We divided the resulting dataset of 42041 chains into (Eukaryotic+Virus) dataset with 21806 chains and Prokaryotic dataset with 20203 chains, with the help of organism classification annotation retrieved from their respective UniProt (Universal Protein Resource) files. UniProt is a central repository of protein sequences and annotation data.

In order to remove redundancy in terms of homology between the chains, we used BLAST-p program on each sequence of the 2 datasets and clustered the sequences with a sequence identity > 25%. BLAST stands for 'Basic Local Alignment Search Tool' [Altschul et al, 1990], and is a set of programs, used to finds regions of local similarity between sequences (protein or nucleotide). The program compares nucleotide or protein sequences to sequence databases and calculates the statistical significance of matches. BLAST-p (also known as protein-protein BLAST) is a specific version of BLAST used for both identifying a query amino acid sequence and for finding similar sequences in protein databases. Sequence identity of 25% was chosen because of a general understanding, which is based on a conservative estimate

that sequences having more than 25% identical residues have similar 3D conformation.

Further, we refined the clusters by removing sequences with sequence length less than 50, and chains belonging to viruses (as their complete 'taxonomic classification' is still unknown, so as to say no one is sure if they belong to Eukaryotes or Prokaryotes). We picked the longest sequence from each cluster to make the non-homologues dataset.

Further, we used Sequence Retrieval System (SRS) hosted by European Bioinformatics Institute (EBI) to download the cysteine disulphide bonding state annotation of whole PDB database. We have adopted theses annotations to label the bonding state of all the cysteines residues present in our Eukaryotic & Prokaryotic datasets.

We also excluded all the inter-bonded (disulphide bonding between two cysteine residues of two different chains), metal bonded (cysteine that coordinate with metal atoms) and redox (cysteine which coordinates with other residues for a short duration of time in order to activate a biological process) cysteine residues from the dataset, and considered only intra-bonded (disulphide bonding between two cysteine residues of same chains) cysteine residues as disulphide bonded cysteine residues in our dataset. Redox cysteine annotation was retrieved from respective UniProt files. For retrieving metal-bonded cysteines, we computed the Euclidian distance between 3D coordinates of SG (Sulphur) atom of cysteine residue and all the atoms of any metal containing compound present in the respective PDB structure. We excluded all those cysteines, which had a metal containing compound with in the radius of 4 Angstrom around their SG atom's 3D space.

A detailed description of the final dataset is provided in below tables 3.1 and 3.2.

Table 3.1. Dataset description on the basis of type of cysteine residues. For each category the number of cysteine residues and their relative percentage is given.

Dataset / Cysteine residues	Eukaryotes	Prokaryotes
Total	4320	3640
Bonded	1496 (35 %)	426 (12 %)
Free	2824 (65 %)	3214 (88 %)

Table 3.2. Dataset description on the basis of type of protein chains. For each category the number of protein chains and their relative percentage is given.

Dataset / PDB Chains	Eukaryotes	Prokaryotes
Total	1041	1329
With both bonded and free cysteine residues	71 (7 %)	36 (3 %)
With only bonded cysteine residues	207 (20 %)	122 (9 %)
With only free cysteine residues	763 (73 %)	1171 (88 %)
With only one cysteine residue (free)	214 (21 %)	431 (32 %)

As we can see from the above tables we have 21 % of chains in Eukaryotic dataset and 32 % chains in Prokaryotic data set with just a single cysteine residue, which cannot make any intra-chain disulphide bridge, and therefore will remain in free states. Prediction of these chains is trivial. However they carry some information of being in free state. Therefore we divided our datasets in to Whole Dataset (WD)

and Reduced Dataset (RD), former including and later excluding single cysteine chains (Table 3.3).

Table 3.3. Dataset description on the basis of type of dataset: Whole Dataset (WD) and Reduced Dataset (RD). For each category number of bonded, free and total number of cysteine residues is given.

Dataset	Eukaryotes (Cysteine residues)	Prokaryotes (Cysteine residues)
WD	Total = 4320 Bonded = 1496 Free = 2824	Total = 3640 Bonded = 426 Free = 3214
RD	Total = 4106 Bonded = 1496 Free = 2610	Total = 3209 Bonded = 426 Free = 2783

3.2. Cross-validation procedure

The most important property of any predictor is its capability to generalize the rules it learns from the training dataset. A predictor, which is able to classify correctly the training dataset but is unable to generalize the learned rules on a new testing set is said to be gone under over fitting This can happen if the training set does not have enough representatives.

In order to test the degree of generalization of the predictor, various methods have been used in previous works (chapter 1, section 1.4). However the basic idea of all the methods is to use a testing set which is disjoint from the training set. One of the procedure known as '*Jack-Knife*' procedure separates one example from the whole dataset each time for testing and keeps remaining for the training, making it a *N-fold* cross validation procedure, where *N* is the no of examples in the whole dataset. This procedure is considered to be the best, as it assures the best estimate of predictor's

performance, but it becomes computationally complex for a large no of *N*. This procedure is also called as '*one-leave-out method*'.

Another method called '*k-fold*' cross validation, where we divide our whole dataset into '*k*' subsets with equal number of representatives in each subset. Cyclically, we test each subset on the basis of the model learned from the training of remaining subsets.

For this thesis work we divided our datasets in to 20 subsets, so as to make it a *20-fold* cross validation procedure.

3.3. Feature Encoding (input coding)

A machine learning method requires some information based on features of the dataset (in our case cysteine examples) as an input in order to learn how to discriminate between labeled classes (in our case 2 classes, Bonded and Free cysteines). For this thesis work, these features are based on global and local properties of the respective cysteine examples. Global properties refer to the properties based on the whole protein sequence of the respective cysteine example, for e.g., length of the protein chain, total number of cysteine residues in the chain. Local properties refer to the properties based on the local environment of the cysteine residue, for e.g., composition of residues, and physiochemical properties of the residues in the local environment of the target cysteine.

For encoding local information, we need to consider a local environment of the cysteine residue of our interest. Computationally this local environment can be considered as a window or sub-string of the whole chain centered at the cysteine of our interest. In this work we have used a window of 27 residues, as we tested the performances of our prediction methods (based on the statistical index described in section 3.5) on different window sizes and found 27 to be the best. The local properties will be based on these 27 residues. We will see below different input features we have used for our predictor.

3.3.1. Single Sequence information

Single sequence information can be considered as the most basic input feature. As local information, it tells the predictor about the presence of a particular residue at a particular position in a given length of window. This information can be encoded by using a vector of 20 (as we have 20 standard residues), putting '100' for the presence of a particular residue and '0' for the remaining 19 residues. For a window size of 27 residues (centered on the cysteine of our interest) this method will lead to a total of 20 * 27 = 540 values. Therefore, 540 values of '0' or '100' will encode single sequence information for each cysteine example.

3.3.2. Sequence Profile composition

Evolutionary information tells the frequency of the presence of a particular residue at a particular position in a chain with respect to the closely related sequences, which may perform same function. This approach generally increases the quality of prediction methods [Fariselli et al. 1999].

Sequence profiles were build in a 2-step procedure:

i) Performing BLAST-p for all the chains present in our datasets against UniProt Knowledgebase database including both reviewed entries from Swissprot and non-reviewed entries from TREMBL databases. This steps searches for the similar sequences in the UniProt Knowledgebase database.

ii) From each of the resultant BLAST-p files of step one, all the sequences reporting an alignment of > 25 % identity are collected and the frequency of each type of amino acid at each position of the chain is computed.

This information was also encoded by 540 values as explained in section 3.3.1. The only difference is that these values are not just '100' or '0', instead they real numbers >=0 and <=100. This is because; sum of 20 standard residues frequencies along with the 'Gap' value (for insertion/deletion of residues in the

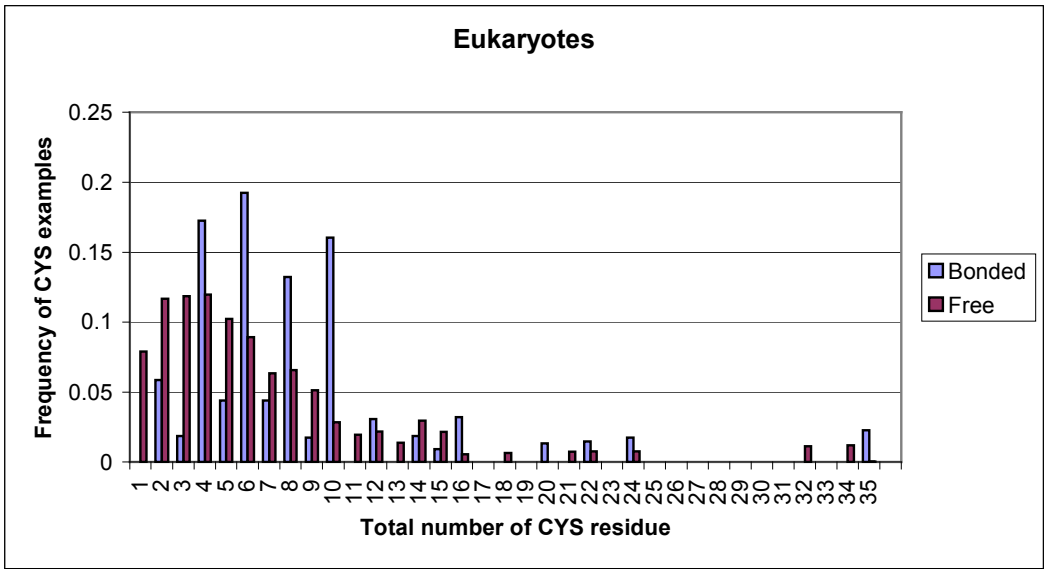
sequence alignment) for each residue position is 100. Additionally, an extra value containing 'Gap' information for each residue position during the sequence alignment in profile files can also be used. This will make a total of 21 * 27 = 567 values to encode for each cysteine example.

3.3.3. Number of cysteine residues

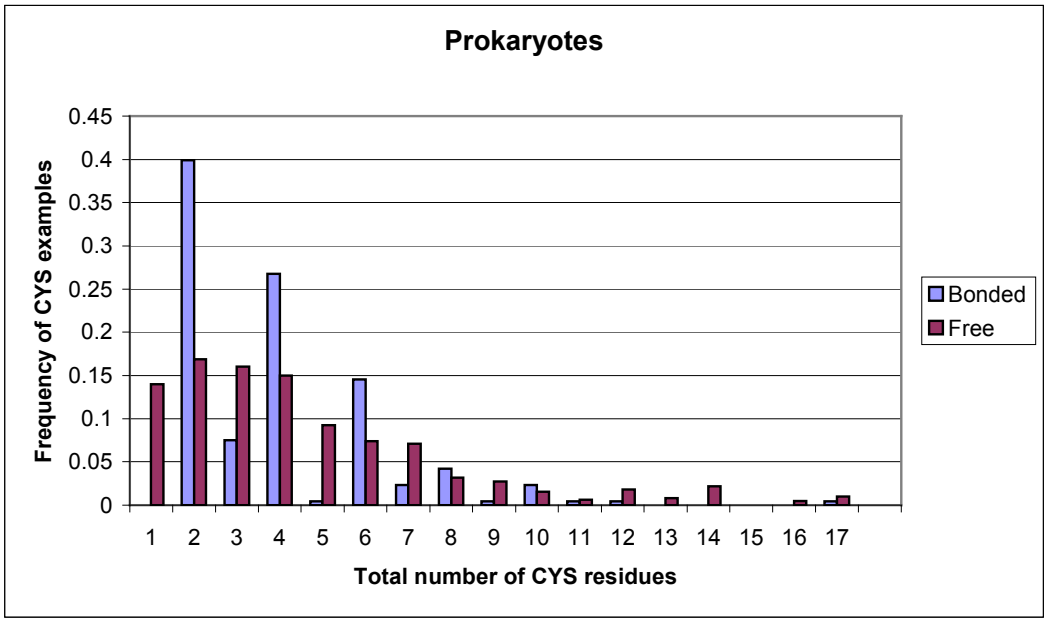
On top of profile information, we can add other global information about the protein chain. And one of this information is the total number of cysteine residues present in the respective protein chains of cysteine examples in our dataset.

This feature can be informative because, cysteines that are present alone in their respective chains will always be in free form (Fig. 3.1a and 3.1b), and chains having only bonded-cysteine examples will have an even number of total number of cysteine residues (obviously a pair of cysteine residue makes a disulphide bond).

This information was encoded by a single value (number of cysteine residues in the chain). A further discussion to encode the same information in a different way is done in the next section 3.3.4.



(a)

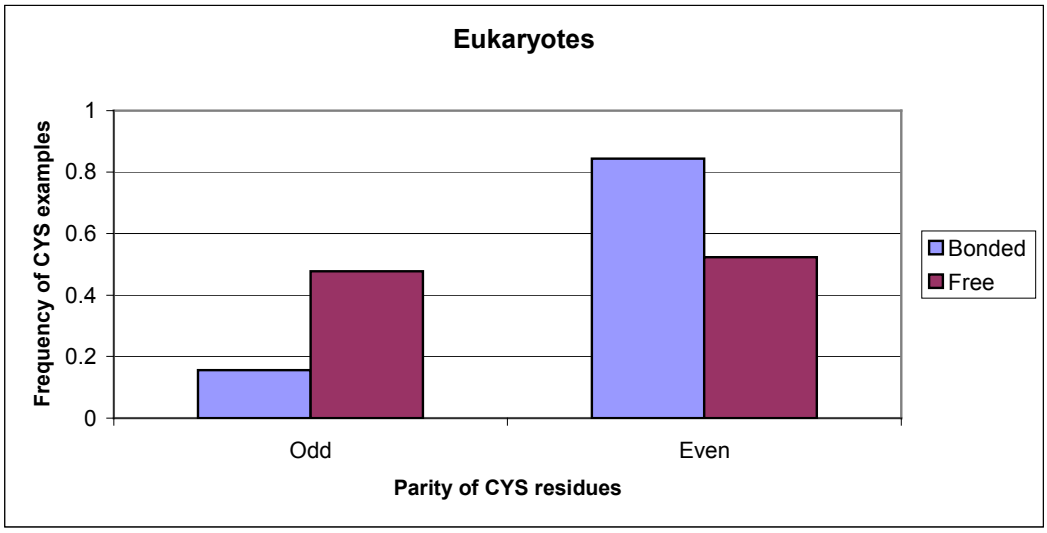


(b)

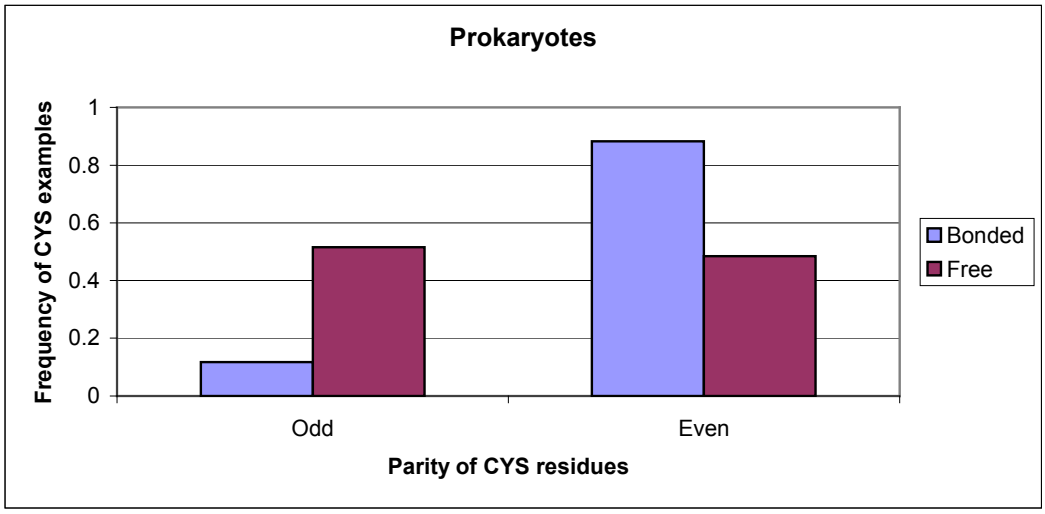
Figure 3.1. Comparison of the distribution of bonded versus free cysteine examples with respect to the total number of cysteine residues present in their respective chains. For both in (a) Eukaryotic WD and (b) Prokaryotic WD, majority of bonded cysteine example chains have an even total of cysteine residues in their respective chains, and majority of free cysteine examples have an odd total of cysteine residues in their respective chains. And cysteine examples, which were present alone in their respective chains, belong to free class.

3.3.4. Parity of cysteine residues

This is another global information, which tells weather the number of cysteine residues in a given chain is even or odd. This feature is also encoded by a single value, '100' for even and '0' for odd number of cysteine. In Fig. 3.2a and 3.2b we see that majority (> 80 %) of bonded cysteine residue examples have an even total of cysteine residues in their respective chains both in Eukaryotes and Prokaryotes. Instead, free cysteines residue examples are found to have roughly equal distribution among their respective chains having even or odd total number of cysteines.



(a)



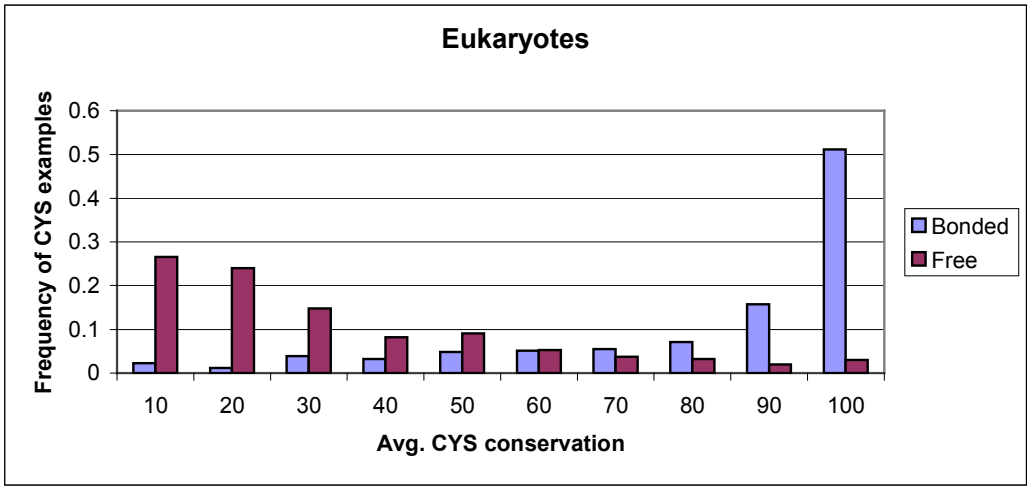
(b)

Figure 3.2. Comparison of the distribution of bonded versus free cysteine examples with respect to the parity of cysteine residues present in their respective chains. For both (a) Eukaryotic WD and (b) Prokaryotic WD, bonded cysteine examples tend to have an even total of cysteine residues in their respective chains. Instead, free cysteine examples tend to have an equal distribution among their respective chains having even or odd total number of cysteines.

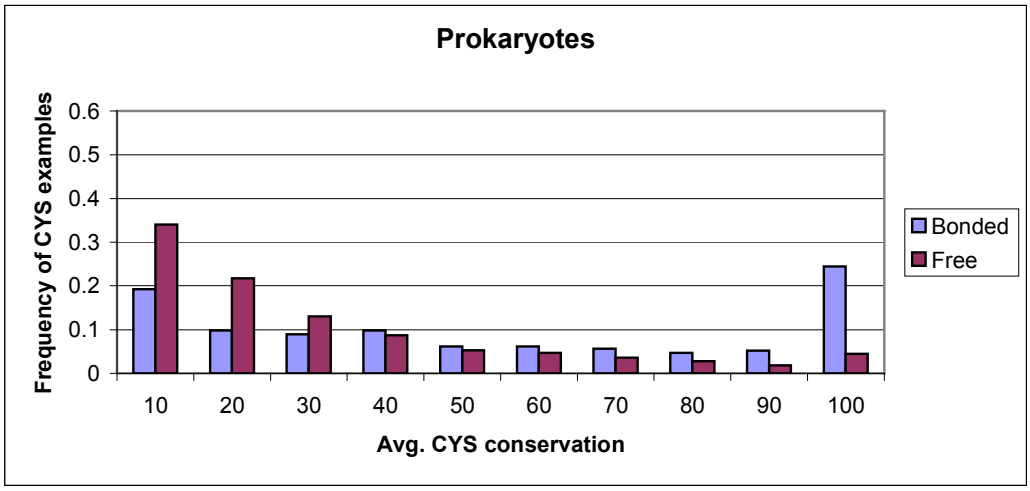
3.3.5. Average Cysteine Conservation

For a given chain, adding the frequencies of all the cysteine residues from profiles and dividing them by the total number of cysteine residues, gives average cysteine conservation value. And with the Fig. 3.3a and 3.3b, we see that majority of bonded cysteines have a high conservation value, confirming their already known importance in protein structure and function (as discussed in chapter 1, section 1.3). However in case of Prokaryotes there is a set of bonded cysteine residues, which have a very low cysteine conservation value. These bonded cysteine examples are difficult to predict (a discussion follows in chapter 4, section 4.4d).

This feature was encoded by a single value >=0 and <=100, as computed by the method discussed above.



(a)



(b)

Figure 3.3. Comparison of the distribution of bonded versus free cysteine examples with respect to the average cysteine conservation. For both (a) Eukaryotic WD and (b) Prokaryotic WD, bonded cysteine examples tend to have a high average conservation value. Instead, free cysteine examples tend to have a very low average conservation value.

3.3.6. Correlated mutation in cysteine residues

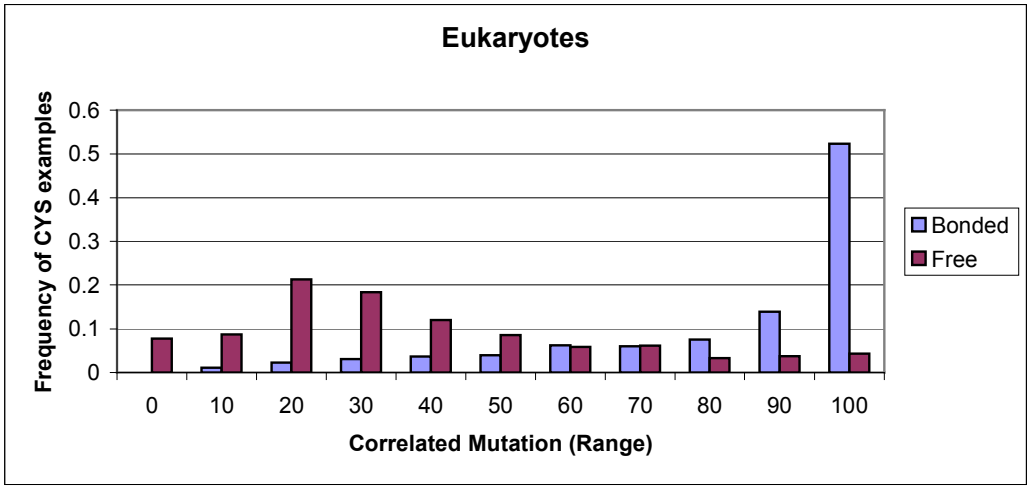
The tendency of residue positions in proteins to mutate coordinately is called Correlated Mutation [Pazos et. al. 1997].

From the BLAST-p alignment of chains, for each cysteine residue we computed its frequency of being correlatively mutated with respect to all other cysteine residues present in the same chain, by counting the number of times two cysteine residue are either present together or absent together and dividing it by the total number of count. For each cysteine example, we took the maximum frequency value computed (as explained above) with respect to all the cysteine residues present in the respective chain, as its correlated mutation value.

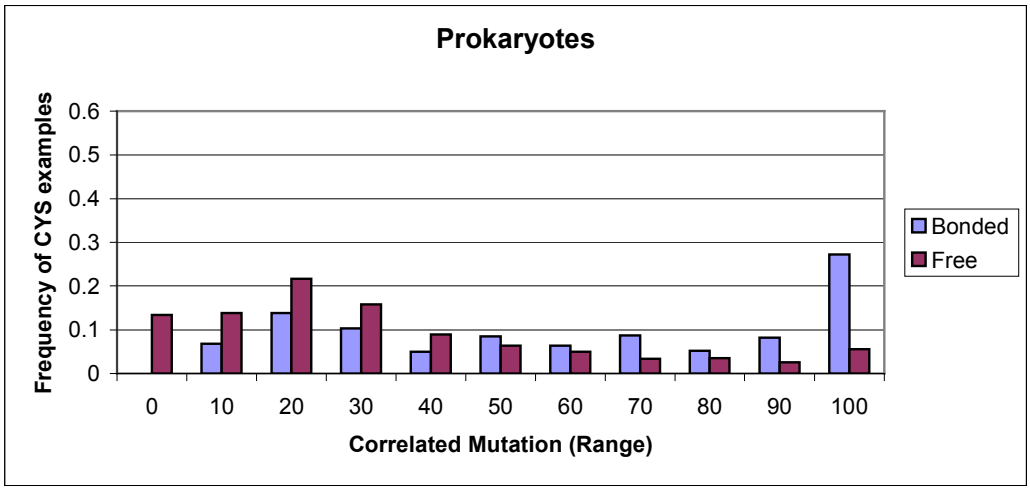
In Fig. 3.4a (for Eukaryotic WD) and 3.4b (for Prokaryotic WD), we see that majority of bonded cysteines examples have high correlated mutation value, which confirms the already known structural and functional importance of these cysteine residues in proteins of the same family and in their evolution (as discussed earlier in chapter 1, section 1.3). However, in case of Prokaryotes (Fig 3.4b) there is a sub-set

of cysteine examples, which in spite of being bonded have a very low correlated mutation value, and these examples got wrongly predicted. These are the same sub-set of bonded examples, which showed low average cysteine conservation (Fig. 3.3.b). These bonded cysteine examples are difficult to predict (a discussion follows in chapter 4, section 4.4e).

This feature was encoded by a single value >=0 and <=100, as computed by the method discussed above.



(a)



(b)

Figure 3.4. Comparison of the distribution of bonded versus free cysteine examples with respect to the correlated mutation. For both (a) Eukaryotic WD and (b) Prokaryotic WD, bonded cysteine examples tend to have a high correlated mutation value. Instead, free cysteine examples tend to have a very low correlated mutation value.

3.3.7. Sub-Cellular Localization

Proteins with disulphide bonded cysteine residues are majorly found in extra-cellular environment of cell. And the reason for this we know is, most of the cellular compartments have a reducing environment, which do not favor a disulphide bond formation, and therefore cysteine residues here tend to remain in free state. Contrary, the oxidizing extra-cellular environment favors the disulphide bond formation, and therefore cysteine residues here are mostly found in bonded state. However, as always some exceptions exist in Biology [Mallick et al. 2002, Riemer et al. 2009].

In Fig 3.5 below, which is based on the predictions of 'BaCelLo- Balanced subCellular Localization' predictor [Pierleoni et al., 2006 & 2007] shows that majority (~ 90 %) of the proteins of our Eukaryotic WD with bonded cysteine examples got predicted with 'Secretory' annotation, and majority of the proteins with free cysteine examples got predicted with 'Cytoplasm' annotation, confirming the general understanding of sub-cellular localization of proteins with disulphide bonded cysteines, as discussed above.

We used these BaCelLo predictions as an input feature for our Eukaryotic dataset, instead of the experimental annotation from the respective UniProt files of their proteins because the majority of the UniProt files lack 'Sub Cellular Localization' annotation.

This feature was encoded by a vector of 5 (since we have 5 types of sub-cellular localization annotation predicted by Bacello), by putting '100' for the presence of a sub-cellular localization and '0' for the remaining 4 values.

In case of Prokaryotes, since they lack cellular compartments, we cannot have annotation for the sub-cellular localization. However, another way to capture this information is the presence of 'Signal Peptides' discussed in next section 3.3.8.

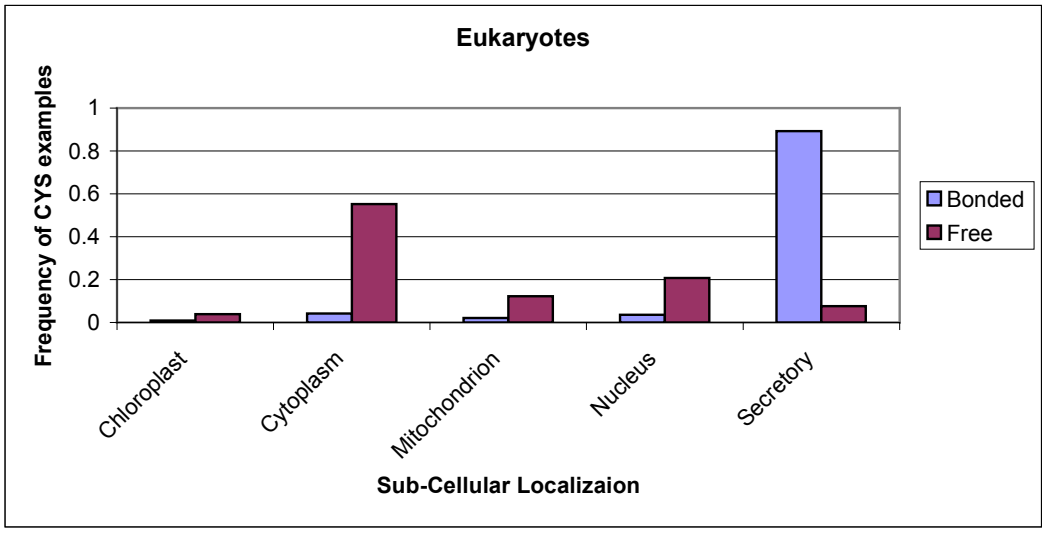


Figure 3.5. Comparison of the distribution of bonded versus free cysteine examples of Eukaryotic WD with respect to the sub-cellular localization (as predicted by BaCelLo) of their respective chains. Majority of bonded cysteine examples belong to secretory proteins (found in extra-cellular region). Instead, majority of free cysteine examples belong to intracellular proteins.

3.3.8. Signal peptide

A Signal peptide is a short peptide chain made up of 3-60 amino acid residues that directs the transport of the protein to other cell organelles. In case of prokaryotes, since they lack cell organelles, these protein chain get directed to extra-cellular region, which has an oxidizing and stable environment. And as we discussed earlier, cysteine residues of these proteins in the extra-cellular environment form disulphide bonds to give stability to the structure of the protein, which is eventually responsible for the specific function of the protein.

In Fig. 3.6, which is based on the signal peptide predictions given by the 'SPEPLip - Predictor of Signal Peptide and Lipoprotein Cleavage Sites in Proteins' [Fariselli et al. 2003], we can see that majority (~ 96 %) of the Prokaryotic free cysteine residues lacked signal peptide in their respective protein chains, however approximately 60% of the bonded cysteine residues have signal peptide in their respective protein chain.

We used these SPEPLip predictions as an input feature for my Prokaryotic dataset, instead of the experimental annotation from the respective UniProt files of their proteins because the majority of the UniProt files lack signal peptide annotation.

This feature is encoded by a single value, ‘100’ for the presence of a signal peptide and ‘0’ for the absence of the signal peptide.

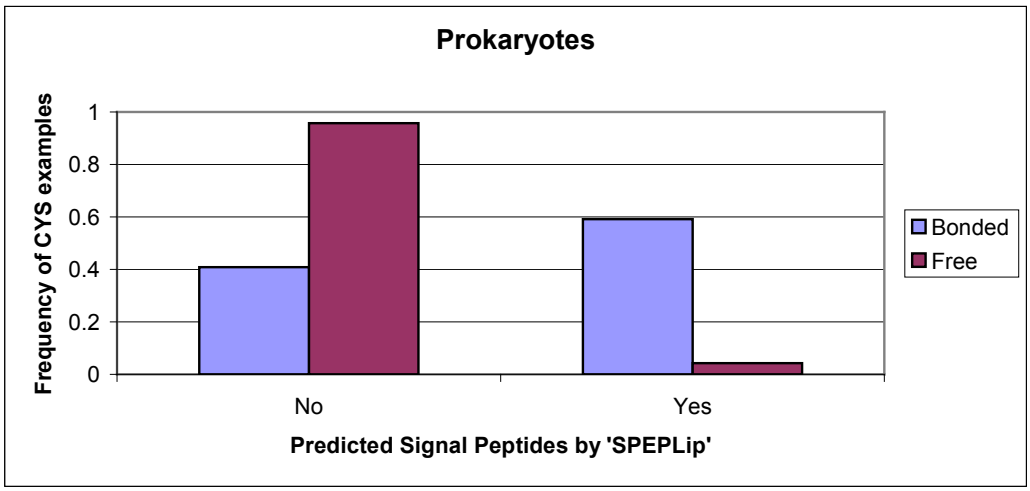


Figure 3.6. Comparison of the distribution of bonded versus free cysteine examples pf Prokaryotic WD with respect to the presence of signal peptide (as predicted by SPEPLip) in their respective chains. Approximately 60 % of chains of bonded cysteine examples tend to have a signal peptide (i.e. are found in extra-cellular region). Instead, majority (~ 96 %) of chains of free cysteine examples tend to not have a signal peptide (i.e. are found in intra-cellular region).

3.4. Methods

We have used two machine-learning techniques, namely 'Artificial Neural Networks (ANN)' and 'Hidden Neural Network (HNN)' for developing our prediction methods.

3.4.1. ANN based predictor

We implemented a standard feed-forward neural network (described in chapter 2, section 2.3) with a back-propagation algorithm (described in chapter 2, section 2.3.1) as a learning procedure. The network architecture (Fig. 3.7) is similar to that used previously [Fariselli et al., 1999 and Martelli et al., 2002a and 2002b] and consist of a two-layer perceptron with two hidden neurons, one output node for discriminating the disulphide bonded and free cysteine propensities respectively, and an input layer that consists primarily of 540 neurons (for 27 residue long window with profile or single sequence input information). Extra neurons were added on top of profile information to add the other global features (described in section 3.3). Table 3.4 describes the total number of input neurons used to encode each feature used for training and testing the ANN predictor.

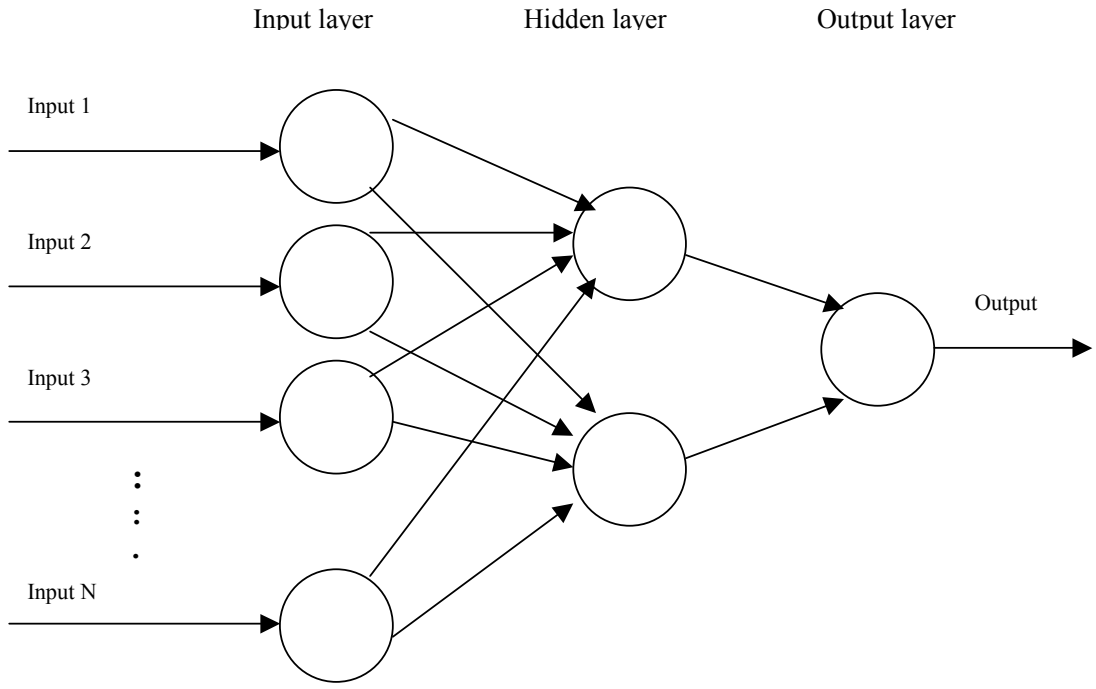


Figure 3.7. Feed-forward artificial neural network architecture used in this thesis work. Input layer consists of *N* number of neurons based on type of input features used (table 3.4). Hidden layer consist of two neurons. And output layer consist of one neuron.

Table 3.4. Description of Input features used as an input for training and testing ANN and HNN based predictors. For each type of input feature, total number of input neurons used for ANN is given.

Method No.	Input Feature	No of input neurons (N)
1.	Single sequence	540
2.	Profiles	540
3.	Profiles + 'Gap' (Insertion/Deletion) information	567
4.	Profiles + Parity of cysteine residues in chains	541
5.	Profiles + Total no. of cysteine residues in chains	541
6.	Profiles + Average cysteine conservation	541
7.	Profiles + Correlated Mutation	541
8.	Profiles + Sub-cellular localization (for Eukaryotes) Profiles + Signal peptides (for Prokaryotes)	545 541
9.	Input features (2+3+4+6+7+8) (for Eukaryotes) Input features (2+3+4+6+7+8) (for Prokaryotes)	575 571

An early learning-stopping procedure was used to train the network [Fariselli et al., 1999]. Also in order to assess the degree of generalization of the prediction method, we used a 20-fold cross validation method (as described in chapter 3, section 3.2).

Performances of the predictor were computed on basis of statistical indexes described in following section 3.5.

Results obtained for both Eukaryotic and Prokaryotic datasets, based on this ANN method and different input features described in table 3.4 have been discussed in the next chapter 4.

3.4.2. HNN based predictor

A vector-based HMM that can handle emission probability vectors, is used on top of the neural networks (described in section 3.4.1). The hybrid system is defined as 'Hidden Neural Network (HNN)' [Krogh and Riis, 1999].

Briefly, considering L as the total number of cysteine residues in the protein chain and A as the size of the alphabet over which vectors are built (i.e. A = 2, bonding and free cysteine states), a sequence vector can be referred with following notation

$$\boldsymbol{s = s^1 s^2 \ldots s^L = [s^1(1), s^1(2)]\ [s^2(1), s^2(2)]\ \ldots\ [s^L(1), s^L(2)]}$$

The components of each vector $\boldsymbol{s^t}$ are positive and sum to a constant value S (independent of position t).

The HMM model (Fig 3.8) used for this thesis work consists of N states connected by means of the transition probabilities a_{ij}. The probability density function for the emission of a vector from each state is determined by a number A of parameters that are peculiar for each state k and are indicated with the symbol $e_k(c)$ (with c = 1,2, … , A):

$$\boldsymbol{P(s^t|\pi^t = k) = (1/Z)\Sigma_c s^t(c) \times e_k(c)}$$

Where π^t is the tth state in the path. Z is the normalizing factor with $\Sigma_c\, e_k(c) = 1$

The vector s^t is obtained directly from the neural network outputs as:

$$\boldsymbol{s^t = [NN(B,W), NN(F,W)]}$$

Where W is the local context of the cysteine and $NN(B,W)$ and $NN(F,W)$ are the neural network estimated probabilities of being in bonding (B) and free (F) state, respectively.

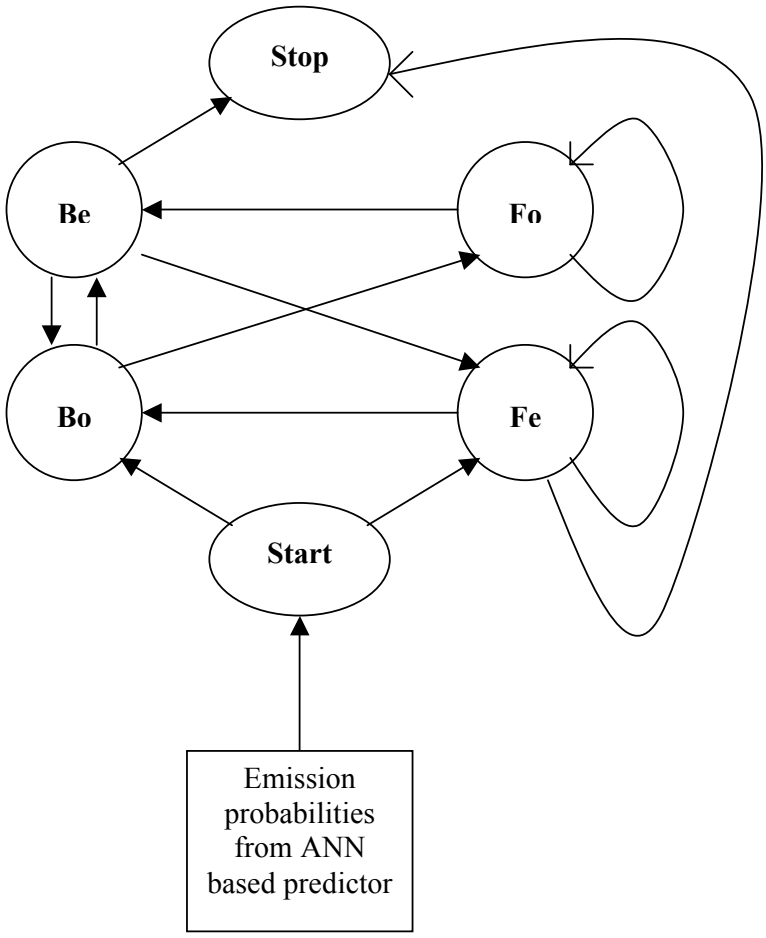


Figure 3.8. HNN state architecture. The arrows represent the allowed transitions. The 'B' and 'F' represent the bonding and free cysteine states, respectively. The label 'e' (even) and 'o' (odd) indicate the number of cysteines in bonding states so far processed. The path can end only from even state, which guarantees that only correct even predictions are assigned when considering intra-chain disulphide bonds.

In order to assess the degree of generalization of the prediction method, a similar method (as used for ANN in section 3.4.1 above) of 20-fold cross validation was used. Training of HMM parameters is accomplished by using a modified expectation-maximization algorithm [Martelli et al., 2002c] In order to keep the constraint derived from HMM model (Fig. 3.8), the prediction of each cysteine is made using Viterbi decoding (as described in chapter 2, section 2.4.1). Performances of the predictor were computed on basis of statistical indexes described in following section 3.5. Results obtained for both Eukaryotic and Prokaryotic datasets, based on

this HNN method and different input features (described in table 3.4) have been discussed in the next chapter 4.

A vector based HMM similar to that used in this work has been applied for the prediction of Beta-barrel proteins [Martelli et al., 2002c] and in an earlier work of prediction of disulphide bonding state of cysteines [Martelli et al., 2002a and 2002b].

3.5. Measure of Performances

The efficiency of predictors (ANN and HNN) was scored using the statistical indices defined as follows.

The accuracy is,

$$\mathbf{Q2 = P/N}$$

Where P is the total number of correctly predicted cysteines and N is the total number of cysteines.

The correlation coefficient C is defined as,

$$\mathbf{C(s) = [p(s) * n(s) - u(s) * o(s)] / \{[p(s) + u(s)][p(s) + o(s)][n(s) + u(s)][n(s) + o(s)]\}^{1/2}}$$

Where for each class s (free or bonded cysteines), $p(s)$ and $n(s)$ are the total number of correct predictions and correctly rejected assignments, respectively and $u(s)$ and $o(s)$ are the number of under-prediction and over-predictions, respectively.

The accuracy for each discriminated class (Bonded and Free cysteines) is evaluated as,

$$\mathbf{Q(s) = p(s)/[p(s)+u(s)]}$$

The probability of correct predictions for each discriminated class s is computed as,

$$\mathbf{P(s) = p(s)/[p(s)+o(s)]}$$

Finally the accuracy per protein is evaluated as,

$$\mathbf{Q2_{prot} = P_p/N_p}$$

Where P_p is the number of the proteins whose cysteines are all correctly predicted and N_p is the total number of proteins.

Chapter 4: Results & Discussion

4.1. Performances of ANN and HNN on Eukaryotes

The ANN-based predictor is to be considered as the basic component of the hybrid system (HNN). Table 4.1a and 4.1b show the results on Eukaryotic reduced dataset (RD) and whole dataset (WD) respectively, using ANN and different types of feature encoding as input (described in table 3.4 and in section 3.3).

Table 4.2a and 4.2b show the results on Eukaryotic reduced dataset (RD) and whole dataset (WD) respectively, using HNN and different types of feature encoding as input (described in table 3.4 and in section 3.3).

From the results described in table 4.1 and 4.2, we can see that profile information improve the performances drastically as compared to the single sequence information. Adding other global features on top of profile information eventually improve the performances significantly. However, the most important features apart from profile information are sub-cellular localization, average cysteine conservation and correlated mutation.

And a combination of all the input features gives the maximum performance.

Table 4.1. ANN-based predictor performances on Eukaryotic RD (a) and WD (b). A 20-fold cross-validation method was applied. Q2 and $Q2_{prot}$ stand for over all accuracy of correct prediction on cysteine basis and protein basis, respectively. Qbo and Pbo are accuracy and probability for correct prediction of bonded cysteine examples, respectively. Qfr and Pfr are accuracy and probability for correct prediction of free cysteine examples, respectively. C is the correlation coefficient. All the statistical indexes were computed based on the formulas described in chapter 3, section 3.5. Results in 'red' colour are the best performances.

(a)

Method No.	Input Feature	Q2	C	Qbo	Qfr	Pbo	Pfr	$Q2_{prot}$
1	Single Sequence	0.73	0.41	0.57	0.82	0.65	0.77	0.42
2	Profiles	0.86	0.71	0.82	0.89	0.81	0.90	0.62
3	2 + Gap	0.87	0.71	0.81	0.90	0.82	0.89	0.62
4	2 + Parity	0.86	0.71	0.84	0.88	0.80	0.91	0.61
5	2 + Total no. of CYS	0.86	0.71	0.82	0.89	0.81	0.90	0.62
6	2 + Avg. CYS Conserv.	0.89	0.75	0.84	0.91	0.84	0.91	0.73
7	2 + Correlated Mutation	0.88	0.74	0.81	0.92	0.85	0.90	0.69
8	2 + Sub Cell Localization	0.92	0.83	0.89	0.94	0.89	0.94	0.83
9	**2 + 3 + 4 + 6 + 7 + 8**	**0.93**	**0.85**	**0.91**	**0.94**	**0.90**	**0.95**	**0.84**

(b)

Method No.	Input Feature	Q2	C	Qbo	Qfr	Pbo	Pfr	$Q2_{prot}$
1	Single Sequence	0.74	0.41	0.57	0.83	0.64	0.79	0.51
2	Profiles	0.86	0.70	0.82	0.88	0.79	0.90	0.66
3	2 + Gap	0.87	0.70	0.81	0.90	0.80	0.90	0.67
4	2 + Parity	0.86	0.70	0.84	0.88	0.78	0.91	0.66
5	2 + Total no. of CYS	0.86	0.70	0.82	0.88	0.79	0.90	0.66
6	2 +Avg. CYS Conserved	0.88	0.73	0.85	0.89	0.81	0.92	0.73
7	2 + Correlated Mutation	0.88	0.74	0.81	0.92	0.85	0.90	0.75
8	2 + Sub Cell Localization	0.92	0.83	0.89	0.94	0.88	0.94	0.85
9	**2 + 3 + 4 + 6 + 7 + 8**	**0.93**	**0.85**	**0.91**	**0.95**	**0.90**	**0.95**	**0.87**

Table 4.2. HNN-based predictor performances on Eukaryotic RD (a) and WD (b). A 20-fold cross-validation method was applied. Q2 and $Q2_{prot}$ stand for over all accuracy of correct prediction on cysteine basis and protein basis, respectively. Qbo and Pbo are accuracy and probability for correct prediction of bonded cysteine examples, respectively. Qfr and Pfr are accuracy and probability for correct prediction of free cysteine examples, respectively. C is the correlation coefficient. All the statistical indexes were computed based on the formulas described in chapter 3, section 3.5. Results in 'red' colour are the best performances.

(a)

Method No.	Input Feature	Q2	C	Qbo	Qfr	Pbo	Pfr	$Q2_{prot}$
1	Single Sequence	0.68	0.25	0.24	0.94	0.68	0.68	0.69
2	Profiles	0.91	0.79	0.85	0.94	0.89	0.91	0.85
3	2 + Gap	0.91	0.80	0.85	0.94	0.89	0.91	0.85
4	2 + Parity	0.90	0.79	0.85	0.94	0.88	0.91	0.85
5	2 + Total no. of CYS	0.91	0.79	0.85	0.94	0.89	0.91	0.85
6	2 +Avg. CYS Conserved	0.90	0.79	0.84	0.94	0.88	0.91	0.85
7	2 + Correlated Mutation	0.91	0.80	0.85	0.94	0.89	0.91	0.85
8	2 + Sub Cell Localization	0.93	0.84	0.88	0.95	0.91	0.93	0.87
9	**2 + 3 + 4 + 6 + 7 + 8**	**0.93**	**0.86**	**0.90**	**0.95**	**0.91**	**0.94**	**0.88**

(b)

Method No.	Input Feature	Q2	C	Qbo	Qfr	Pbo	Pfr	$Q2_{prot}$
1	Single Sequence	0.68	0.26	0.24	0.94	0.68	0.70	0.75
2	Profiles	0.91	0.80	0.85	0.94	0.89	0.92	0.88
3	2 + Gap	0.91	0.80	0.85	0.95	0.89	0.92	0.88
4	2 + Parity	0.91	0.79	0.85	0.94	0.88	0.92	0.88
5	2 + Total no. of CYS	0.91	0.80	0.85	0.94	0.89	0.92	0.88
6	2 +Avg. CYS Conserved	0.91	0.79	0.84	0.94	0.88	0.92	0.88
7	2 + Correlated Mutation	0.91	0.80	0.85	0.95	0.89	0.92	0.88
8	2 + Sub Cell Localization	0.93	0.84	0.88	0.95	0.91	0.94	0.90
9	**2 + 3 + 4 + 6 + 7 + 8**	**0.94**	**0.86**	**0.90**	**0.96**	**0.91**	**0.95**	**0.90**

On adding HMM on top of ANN i.e. using the hybrid system HNN with different input coding improves their performances respectively. Method no. 9 in table 4.2a and 4.2b describes the final performances for Eukaryotes. We reached a remarkable accuracy both on cysteine bases, 93 % for difficult set (RD) and 94 % for whole dataset (WD), and on protein bases, 88 % for difficult set (RD) and 90 % for whole dataset (WD). These accuracies were obtained along with a very high correlation value of 0.86 for both the sets, and are the best so far, when compared with previously developed methods. A comparison of our predictor performances with other previously developed methods is shown in section 4.3 of this chapter.

4.2. Performances of ANN and HNN on Prokaryotes

In case of Prokaryotic dataset a very poor performance has been noted at the basic level of input coding of single sequence and profile information (tables 4.3 and 4.4).

However, with the addition of the other global input features an increase in the Prokaryotic performances was also noted. And best features in case of prokaryotic dataset came out to be parity of cysteine residues and signal peptide information. Even though, on the basis of global statistical indexes of Q2 and $Q2_{prot}$ prokaryotes also performed equally well as of Eukaryotes, but the correlation coefficient (C) value was comparably lesser than as of Eukaryotes. Reason behind the poor performances of prokaryotic dataset as compared to eukaryotic dataset at the primary level of single sequence information and profile information when given as input, and a lower correlation value even after adding other global features has been analysed and discussed in the section 4.4 of this chapter.

Table 4.3. ANN-based predictor performance on Prokaryotic RD (a) and WD (b). A 20-fold cross-validation method was applied. Q2 and $Q2_{prot}$ stand for over all accuracy of correct prediction on cysteine basis and protein basis, respectively. Qbo and Pbo are accuracy and probability for correct prediction of bonded cysteine examples, respectively. Qfr and Pfr are accuracy and probability for correct prediction of free cysteine examples, respectively. C is the correlation coefficient. All the statistical indexes were computed based on the formulas described in chapter 3, section 3.5. Results in 'red' colour are the best performances.

(a)

Method No.	Input feature	Q2	C	Qbo	Qfr	Pbo	Pfr	$Q2_{prot}$
1	Single Sequence	0.85	0.21	0.20	0.95	0.40	0.89	0.72
2	Profiles	0.85	0.38	0.48	0.91	0.44	0.92	0.72
3	2 + Gap	0.86	0.39	0.48	0.92	0.47	0.92	0.72
4	2 + Parity	0.88	0.45	0.47	0.95	0.57	0.92	0.76
5	2 + Total no. of CYS	0.84	0.38	0.54	0.88	0.41	0.93	0.66
6	2 + Avg. CYS Conserved	0.87	0.38	0.41	0.94	0.51	0.91	0.76
7	2 + Correlated Mutation	0.87	0.41	0.44	0.94	0.52	0.92	0.77
8	2 + Signal Peptide	0.92	0.62	0.58	0.97	0.76	0.94	0.89
9	**2+ 3 + 4 + 6 + 7 + 8**	**0.93**	**0.64**	**0.57**	**0.98**	**0.81**	**0.94**	**0.89**

(b)

Method No.	Input feature	Q2	C	Qbo	Qfr	Pbo	Pfr	$Q2_{prot}$
1	Single Sequence	0.87	0.21	0.20	0.96	0.37	0.90	0.80
2	Profiles	0.86	0.36	0.46	0.92	0.42	0.93	0.78
3	2 + Gap	0.86	0.37	0.48	0.92	0.43	0.93	0.78
4	2 + Parity	0.90	0.46	0.47	0.95	0.57	0.93	0.84
5	2 + Total no. of CYS	0.84	0.36	0.54	0.88	0.37	0.93	0.72
6	2 + Avg. CYS Conserved	0.87	0.36	0.41	0.94	0.46	0.92	0.81
7	2 + Correlated Mutation	0.89	0.41	0.44	0.94	0.51	0.93	0.84
8	2 + Signal Peptide	0.92	0.59	0.58	0.97	0.70	0.94	0.90
9	**2 + 3 + 4 + 6 + 7 + 8**	**0.93**	**0.64**	**0.57**	**0.98**	**0.79**	**0.94**	**0.92**

Table 4.4. HNN-based predictor performance on Prokaryotic RD (a) and WD (b). A 20-fold cross-validation method was applied. Q2 and $Q2_{prot}$ stand for over all accuracy of correct prediction on cysteine basis and protein basis, respectively. Qbo and Pbo are accuracy and probability for correct prediction of bonded cysteine examples, respectively. Qfr and Pfr are accuracy and probability for correct prediction of free cysteine examples, respectively. C is the correlation coefficient. All the statistical indexes were computed based on the formulas described in chapter 3, section 3.5. Results in 'red' colour are the best performances.

(a)

Method No.	Input feature	Q2	C	Qbo	Qfr	Pbo	Pfr	$Q2_{prot}$
1	Single Sequence	0.87	0.0	0.0	1.0	0.0	0.87	0.82
2	Profiles	0.89	0.36	0.17	0.996	0.88	0.89	0.85
3	2 + Gap	0.88	0.25	0.09	0.997	0.83	0.88	0.84
4	2 + Parity	0.89	0.43	0.31	0.98	0.75	0.90	0.86
5	2 + Total no. of CYS	0.90	0.44	0.24	0.995	0.90	0.90	0.86
6	2 + Avg. CYS Conserved	0.89	0.43	0.27	0.99	0.81	0.90	0.86
7	2 + Correlated Mutation	0.90	0.44	0.28	0.99	0.82	0.90	0.86
8	**2 + Signal Peptide**	**0.93**	**0.65**	**0.57**	**0.98**	**0.82**	**0.94**	**0.90**
9	2 + 3 + 4 + 6 + 7 + 8	0.92	0.63	0.59	0.97	0.77	0.94	0.90

(b)

Method No.	Input feature	Q2	C	Qbo	Qfr	Pbo	Pfr	$Q2_{prot}$
1	Single Sequence	0.88	0.0	0.0	1.0	0.0	0.88	0.88
2	Profiles	0.90	0.37	0.17	0.996	0.88	0.90	0.90
3	2 + Gap	0.89	0.26	0.09	0.997	0.83	0.89	0.89
4	2 + Parity	0.91	0.44	0.31	0.99	0.75	0.91	0.90
5	2 + Total no. of CYS	0.91	0.44	0.24	0.996	0.90	0.91	0.91
6	2 + Avg. CYS Conserved	0.91	0.43	0.27	0.99	0.81	0.91	0.91
7	2 + CorrMutOcc	0.91	0.45	0.28	0.99	0.82	0.91	0.91
8	**2 + Signal Peptide**	**0.94**	**0.65**	**0.57**	**0.98**	**0.82**	**0.95**	**0.93**
9	2 + 3 + 4 + 6 + 7 + 8	0.93	0.64	0.59	0.98	0.77	0.95	0.93

Prediction at the second level i.e. using the hybrid system HNN showed an improvement in the performances at both cysteine and protein level (Fig. 4.4a and 4.4b). Signal peptide information on top of profiles came out to be the best input for predicting prokaryotes. Method no. 8 in table 4.3a and 4.3b describes the final performances for Prokaryotes. We reached a remarkable accuracy both on cysteine bases; 93 % for difficult set (RD) and 94 % for whole dataset (WD), and on protein bases; 90 % for difficult set (RD) and 93 % for whole dataset (WD). These accuracies were obtained along with a decent correlation value of 0.65 for both the sets, and are the best so far, when compared with previously developed methods. A comparison of our predictor performance with other previously developed methods is shown in section 4.3 of this chapter.

4.3. Comparison of results with previously developed methods

It is difficult to compare methods tested on different databases. However, it can be claimed that the performances obtained when considering Eukaryotes and Prokaryotes separately, and with the incorporation of global features specifically parity of cysteine residues, average cysteine conservation, correlated mutation, sub-cellular localization, and signal peptide, are greater than that previously described and obtained with other methods.

Table 4.5a & 4.5b show a comparison of results of different methods developed in recent years. In the most recent work done by Ceroni et al. in 2006 and Chen et al. in 2004, they have not shown the performances on the difficult set (RD) (table 4.5a). Also in the work of Chen et al., they have not shown performance on protein basis. In the work of Ceroni et al in 2006, they have not shown the most important statistical index of Correlation Coefficient (*C*). It is important to compute this index because it tells how well the method is capable to learn and discriminate between two classes with equal efficiency. In these terms, the only previous work [Martelli et al. 2002a and 2002b] clearly shows the performances on all the possible statistical indexes and allows us for a direct comparison with our results. Our work is principally based on the hybrid method HNN using profile information as a primary input, developed by Martelli et al. in 2002. Addition of more global protein features

on top of profile information, and considering eukaryotic and prokaryotic dataset separately has made our method to outperform all the previous methods.

Table 4.5. Comparison of performances of our final prediction method (as described in table 4.2 and 4.4) with previously developed methods in terms of RD (a) and WD (b). Statistical indexes are the same as described in above tables. *NA* means data 'Not Available'.

(a)

Method	Q2	C	Q(B)	Q(F)	P(B)	P(F)	Q2prot
Martelli et al. (2002)	87.4	0.73	78.1	92.8	86.3	88.0	80.2
Song et al. (2004)	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
Chen et al. (2004)	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
Ceroni et al. (2006)	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*	*NA*
This work on Eukaryotes (2009)	93.0	0.86	90.0	95.0	91.0	94.0	88.0
This work on Prokaryotes (2009)	93.0	0.65	57.0	98.0	82.0	94.0	90.0

(b)

Method	Q2	C	Q(B)	Q(F)	P(B)	P(F)	Q2prot
Martelli et al. (2002)	88.0	0.73	78.1	93.3	86.3	88.8	84.0
Song et al. (2004)	89.1	0.71	92.2	79.3	*NA*	*NA*	85.2
Chen et al. (2004)	90.0	0.77	77.0	97.0	91.0	89.0	*NA*
Ceroni et al. (2006)	88.0	*NA*	*NA*	*NA*	*NA*	*NA*	83.0
This work on Eukaryotes (2009)	94.0	0.86	90.0	96.0	91.0	95.0	90.0
This work on Prokaryotes (2009)	94.0	0.65	57.0	98.0	82.0	95.0	93.0

4.4. Why Eukaryotes perform better than Prokaryotes ?

As we have seen in our result section (tables 4.1 - 4.4), that even though we have reached to approximately equal accuracies in our final methods (in 'red' colour) at both cysteine ($Q2$) and protein level ($Q2_{prot}$), for both eukaryotic and prokaryotic datasets, but still the correlation coefficient (C) value for Prokaryotes is lesser than as compared to that of Eukaryotes. Reason for this is, statistical indexes concerning to bonded cysteines ($Q(B)$ & $P(B)$) in prokaryotes is lesser than that of free cysteine ($Q(F)$ & $P(F)$) examples. Which indicates, that in case of Prokaryotes even though our method learns very well to predict free cysteine examples but do not learns equally well for predicting bonded cysteine example.

Possible reasons for this we found, are discussed below:

a) Highly unbalanced prokaryotic dataset:

Number of bonded cysteine examples (426) in prokaryotic dataset was very less than their number of free cysteine examples (3214), making it a ratio of 1:8. And as discussed earlier in chapter 3 (section 3.2), an unbalanced dataset may lead to over fitting of the results. However a balancing procedure was adopted by repeating the bonded cysteine examples in the training set so as to equalize the ratio of both the classes, but no significant improvement in performances was observed.

However in case of Eukaryotes, this difference between bonded (1496) and free (2824) cysteine examples was in a ratio 1:2 respectively, which was quite low as compared to that in Prokaryotes. Here also we adopted the same balancing procedure, but ended up with no significant improvement in the performances.

b) **Composition of residues in the local environment**

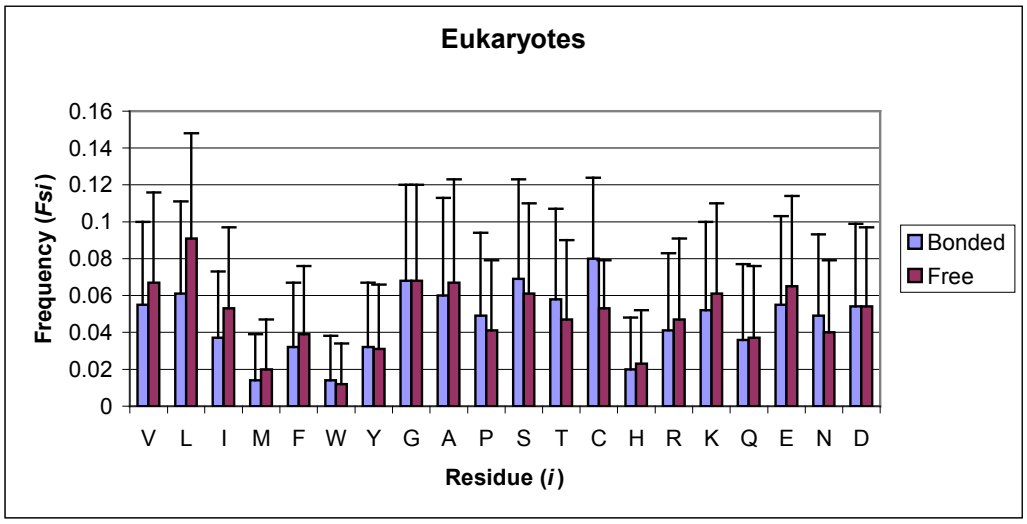
Considering the local environment of 27-residue window centered at the cysteine of our interest in the respective protein chains of bonded and free cysteine examples, we computed frequency Fs_i of each 20 standard residue 'i' by using the formula:

$$\boldsymbol{Fs_i = n_i/N}$$

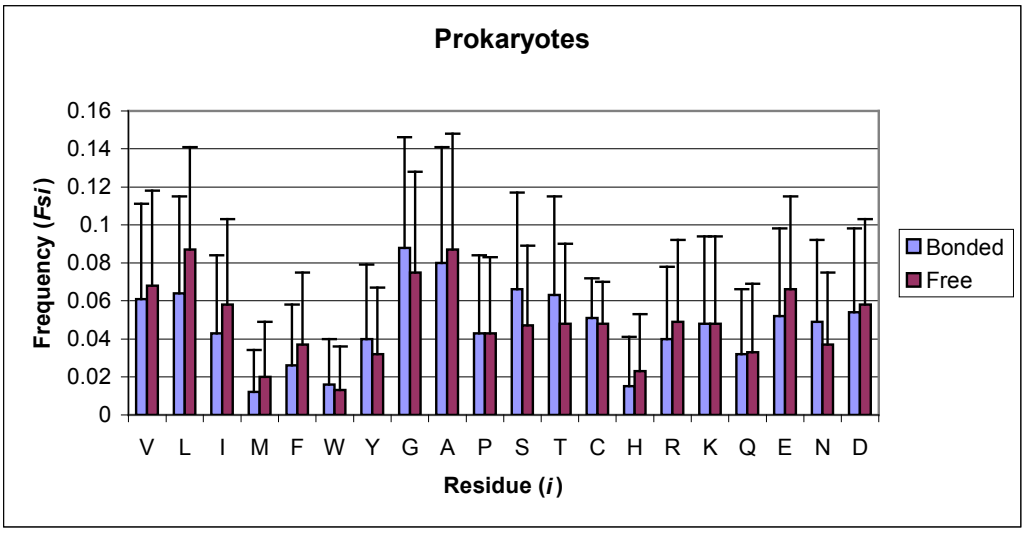
Where '*s*' is the class (bonded or free). 'n_i' is the total number of times the residue '*i*' is present in 27-residue window of '*s*' examples, and '*N*' is the total number of residues in 27-residue window of '*s*' examples.

Eukaryotes show a significant difference in terms of amount of cysteine residue in bonded and free cysteine examples (Fig 4.1a). Eukaryotic bonded cysteine examples tend to have a cysteine rich environment as compared to their free cysteine examples. However, in case of Prokaryotes this difference is approximately zero (Fig 4.2b). This feature seemed to be very important and a further investigation and discussion is done in next section 4.4c.

Apart from Cysteine residue, small differences in amount of 'Glycine', 'Proline', 'Serine', 'Threonine', and 'Glutamic Acid' were also noted. For the rest of the residues the differences in their amount were roughly the same for both Eukaryotes and Prokaryotes.



(a)



(b)

Figure 4.1. Comparison of frequencies of residues in the local (27-residue window) environment of bonded and free cysteine examples of Eukaryotes (a) and Prokaryotes (b). Error bars were plotted by computing the standard error of the mean.

c) **Eukaryotes bonded cysteine examples have a 'symmetric-cysteine-rich' environment**

In order to understand more deeply about the presence of cysteine residues in the local environment (27-residue window) of bonded and free cysteine examples separately, we computed the percentage of amount of cysteine residue (PCs_j) for each 27 positions 'j', by using a formula:

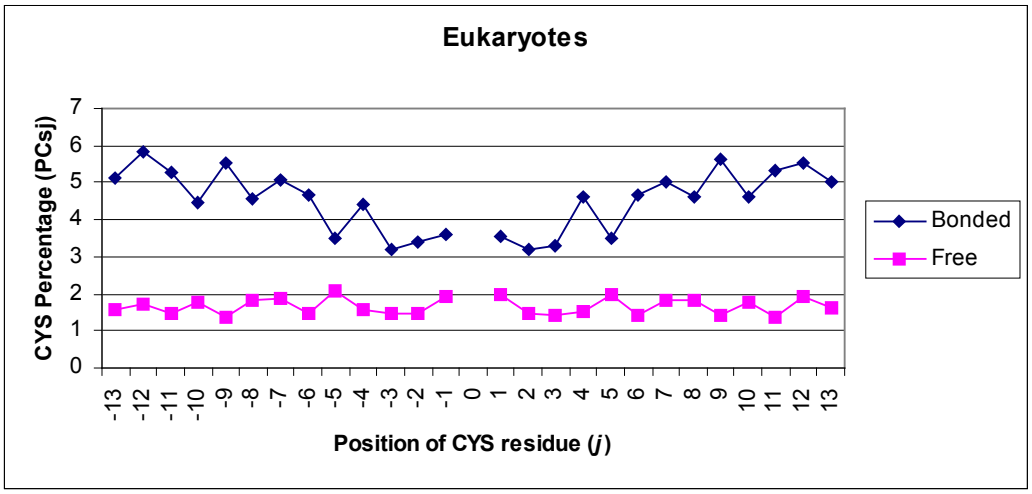
$$PCs_j = ((TCs_j) * 100) / Ns$$

Where 's' is the class (bonded or free). 'TCs_j' is the total number of times a cysteine residue is present at position 'j' in 's' examples. And 'Ns' is the total number of 's' examples.

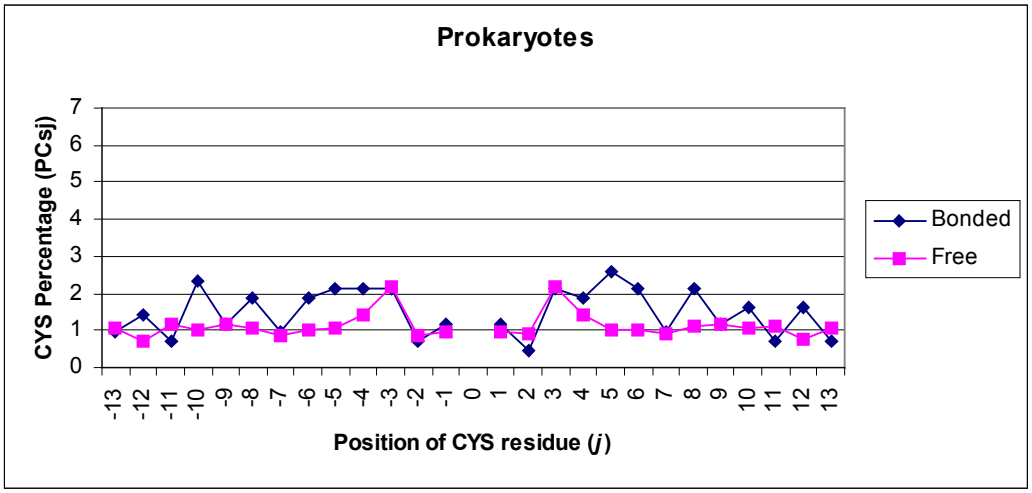
And as per our expectation derived from the Fig. 4.1a above, that Eukaryotes have a significant difference with respect to the amount of cysteine residue in bonded and free cysteine examples, Fig. 4.2a below confirms this difference more clearly with respect to each 27 position of the local environment. This result gives a more clear interpretation that Eukaryotic bonded cysteine examples tend to carry a very rich cysteine environment as compared to their free cysteine examples. And this seemed to be the most important feature learned by our predictor to discriminate between the two classes (bonded and free) based on the basic input of single sequence information or profile information.

Prokaryotes lacked this feature (Fig 4.2b) substantially, and this was one of the reasons behind their poor prediction performances at the basic level. Addition of other global features could not improve the statistical indexes ($Q(B)$, $P(B)$ & C) comparable to Eukaryotes because they were already performing badly at the very basic level.

Another interesting thing about Eukaryotes was that they showed a perfect symmetry (Fig 4.2a) in terms of the amount of the cysteine residue present in both the directions (+13 to –13) of target cysteine residue. However, in case of Prokaryotes (Fig. 4.2b), only few (+10 & –10, +6 & –6, and +5 & –5) were important symmetric positions for bonded cysteine examples.



(a)



(b)

Figure 4.2. Comparison of percentage of cysteine residues in the local (27-residue window) environment of bonded and free cysteine examples of Eukaryotes (a) and Prokaryotes (b).

d) Prokaryotic bonded cysteine residues are poorly conserved

As evident from the comparison of Fig. 3.3a and 3.3b, prokaryotic bonded cysteine examples on an average are less conserved than eukaryotic bonded cysteine examples.

During evolution of proteins, those residues tend to remain conserved, which play an important role in structural and functional properties of proteins. As discussed earlier in chapter 1 section 1.3, Cysteine is an important residue in putting structural constraint on the three-dimensional structure of proteins by making disulphide bridges with another cysteine residue, which ultimately decides a unique fold for the protein. This unique fold or structure is responsible for a unique function of the protein. In case of Eukaryotes (Fig. 3.3a) this rationale comes out to be quite true. However, approximately 50% of bonded cysteine examples of Prokaryotes (Fig. 3.3b) seem to not follow this general rule, and show a poor conservation in their respective profile alignments.

As expected both in Eukaryotes (Fig. 3.3a) and Prokaryotes (Fig 3.3b), majority of free cysteine examples have a very low conservation value in their respective profile alignments, confirming that being in free state they do not

have an important role to play in deciding structure and function of their respective protein.

This feature of 'Average Cysteine Conservation', which had a more clear discrimination between bonded and free cysteine examples in Eukaryotes than in Prokaryotes, was eventually learned better by the ANN predictor in case of testing Eukaryotes (table 4.1a and 4.1b, method no. 6) and gave better performance, than in Prokaryotes (table 4.3a and 4.3b, method no. 6).

e) Prokaryotic bonded cysteine residues have a low correlated mutation value

As evident from the comparison of Fig. 3.4a and 3.4b, prokaryotic bonded cysteine examples tend to have a poor correlated mutation values as compared to Eukaryotic bonded cysteine examples.

In terms of protein evolution, the two-cysteine residues will be present or absent together when they have a coordinated structural role to play. In case of bonded cysteine examples, they will form a disulphide bridge only if both of them are present and therefore will share a high correlation value with respect to each other. On the other hand, free cysteines being structurally unimportant do not share any relationship with any other cysteine residue and thus leading to a very low correlation mutation values.

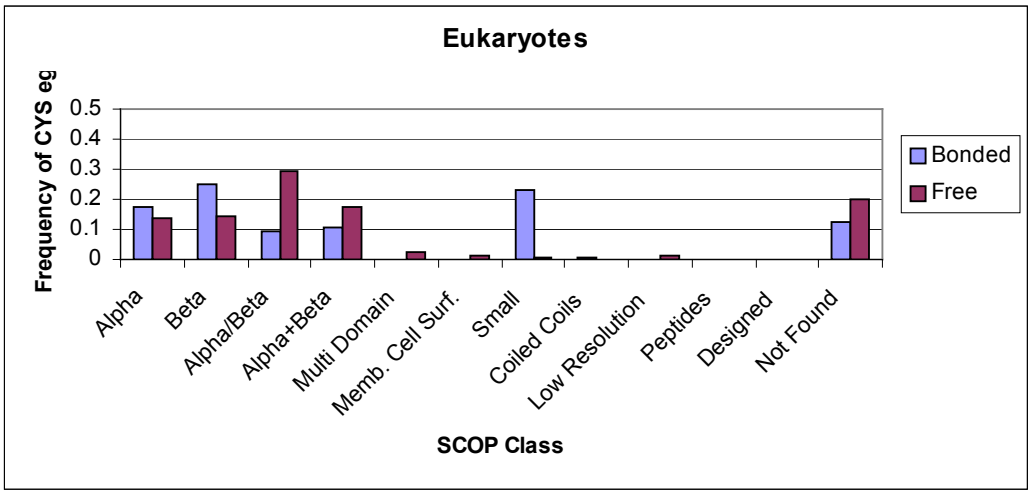
Eukaryotes tend to follow this rationale very well, and therefore have a quite clear discrimination between bonded and free cysteine examples (Fig. 3.4a), and thus it is easier for a predictor to correctly classify them (table 4.1a and 4.1b, method no. 7).

Prokaryotes on the other hand, do not follow this feature well (Fig. 3.4b) and therefore the classification is more difficult, and our predictor shows lower performances in this case (table 4.3a and 4.3b, method no. 7).

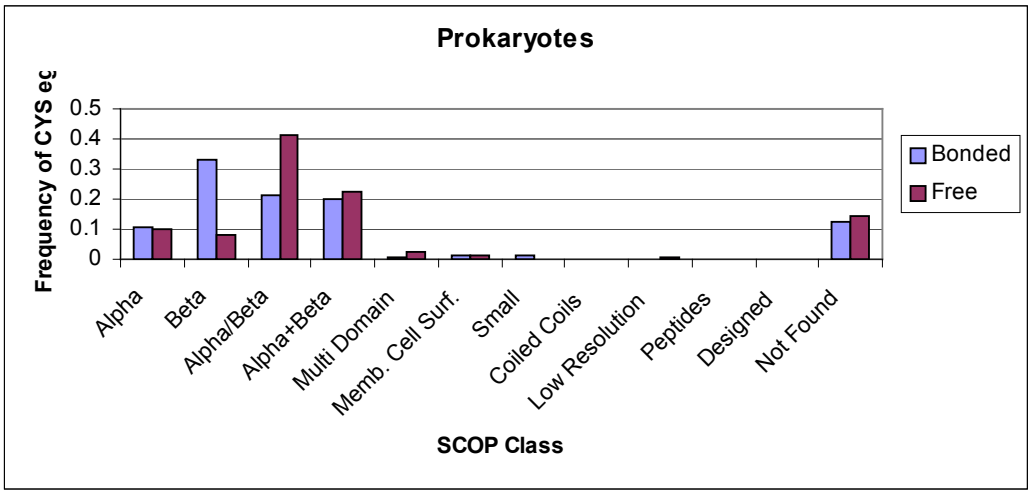
As expected, free cysteine examples, in case of both Eukaryotes and Prokaryotes have a very lower correlated mutation values, confirming their unimportance in structural and functional properties of their respective proteins.

f) Prokaryotes lacked disulphide bond rich 'Small Proteins'

We did a SCOP classification of proteins in our datasets and found that approximately 23.3% of the Eukaryotic bonded cysteine examples belong to 'Small Proteins' class (Fig. 4.3a), as compared to just 1.5% in Prokaryotes (Fig. 4.3b). SCOP stands for (Structural Classification Of Proteins) is an online database which provides a broad survey of all known protein folds and annotates the folding class for the known protein structures present in PDB.



(a)



(b)

Figure 4.3. SCOP classification of cysteine examples. In Eukaryotic dataset (a) 'Small Proteins' have contributed significantly (23.3% of the total bonded cysteine examples), whereas in Prokaryotic dataset (b) their contribution is negligible.

These 'Small Proteins' in Eukaryotes (table 4.6a) were found to be very rich in cysteines with a cysteine/protein ratio of 8.2:1, and also very rich in disulphide bonds with a bonded/free cysteine ratio of 35:1.

Table 4.6. Description of dataset on the basis of SCOP class. In Eukaryotic dataset (a) 'Small Proteins' (in red color) have the highest cysteine/protein ratio of 8.2:1 and highest bonded/free cysteine ratio of 35:1. In Prokaryotic dataset (b) their examples are very few.

(a)

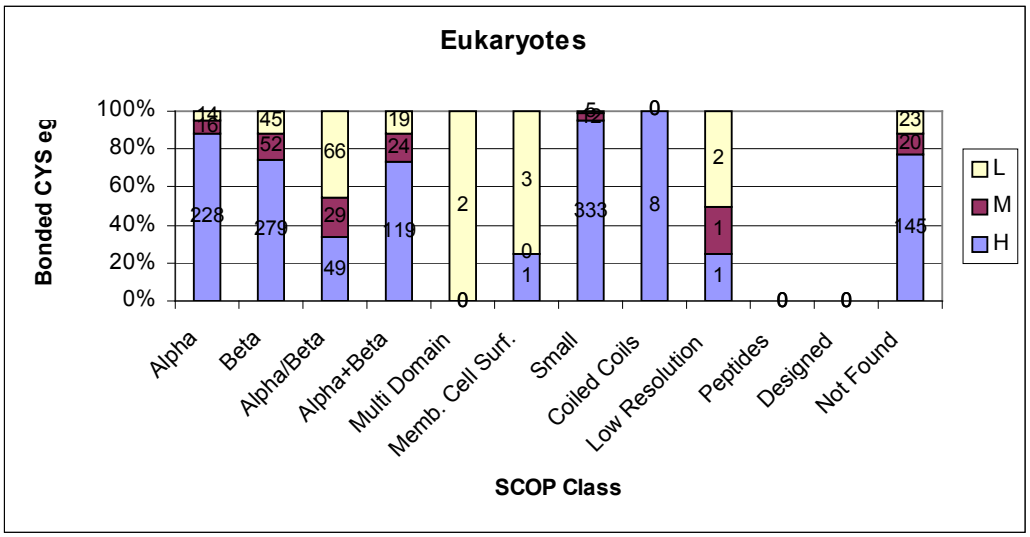
SCOP Class	No. of Proteins	No. of Cysteine	Cysteine/ Protein	Bond	Free	Bonded/ Free
All Alpha	177	652	3.7	258	394	0.7
All Beta	215	775	3.6	376	399	0.9
Alpha/Beta	208	968	4.7	144	824	0.2
Alpha+Beta	219	659	3.0	162	497	0.3
Small	**44**	**360**	**8.2**	**350**	**10**	**35.0**
Multi Domain	12	65	5.4	2	63	0.03
Membrane Cell Surface	9	33	3.7	4	29	0.1
Coiled Coils	4	11	2.8	8	3	2.7
Low Resolution	7	38	5.4	4	34	0.1
Peptides	1	1	1.0	0	1	0
Designed	1	8	8.0	0	8	0
Not Found	199	750	3.8	188	562	0.3

(b)

SCOP Class	No. of Proteins	No. of Cysteine	Cysteine/ Protein	Bond	Free	Bonded/ Free
All Alpha	179	362	2.0	44	318	0.1
All Beta	178	407	2.3	140	267	0.5
Alpha/Beta	472	1418	3.0	90	1328	0.1
Alpha+Beta	360	804	2.2	86	718	0.1
Small	**4**	**12**	**3.0**	**6**	**6**	**1.0**
Multi Domain	23	78	3.4	2	76	0.02
Membrane Cell Surface	19	38	2.0	6	32	0.2
Coiled Coils	0	0	0	0	0	0
Low Resolution	7	13	1.9	0	13	0
Peptides	1	1	1.0	0	1	0
Designed	0	0	0	0	0	0
Not Found	190	507	2.7	52	455	0.1

These bonded cysteines of 'Small Proteins' were found to be very highly conserved in their respective 'Profile' files as compared to other bonded cysteine examples of other SCOP classes (Fig. 4.4a and 4.4b).

(a)



(b)

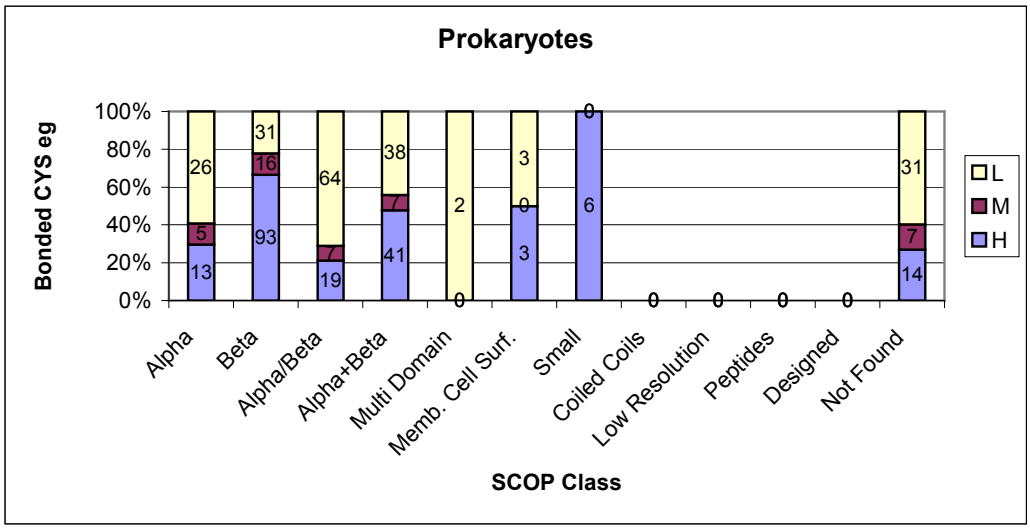
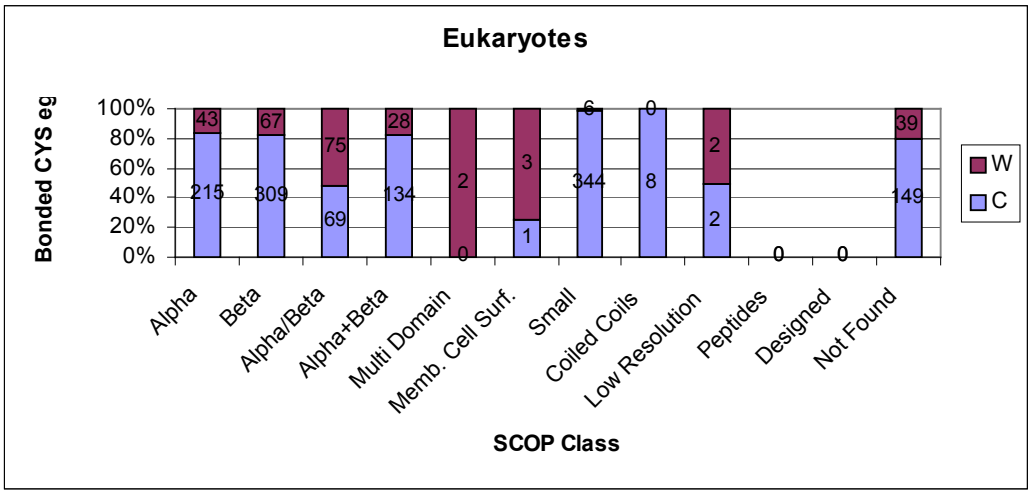


Figure 4.4. Comparison of SCOP classes versus conservation (frequency, 'Fb') of bonded cysteine residues in 'Profiles'. 'L' = Low (0 > Fb <=33), 'M' = Medium (33 > Fb <=66), and 'H' = High (66 > Fb <=100). Bonded cysteine examples belonging to 'Small Proteins' were almost all found to be highly conserved both in Eukaryotes (a) and Prokaryotes (b).

And these bonded cysteines of 'Small Proteins' were getting predicted correctly with a very high accuracy of 98% in case of Eukaryotes (Fig. 4.5a).

(a)



(b)

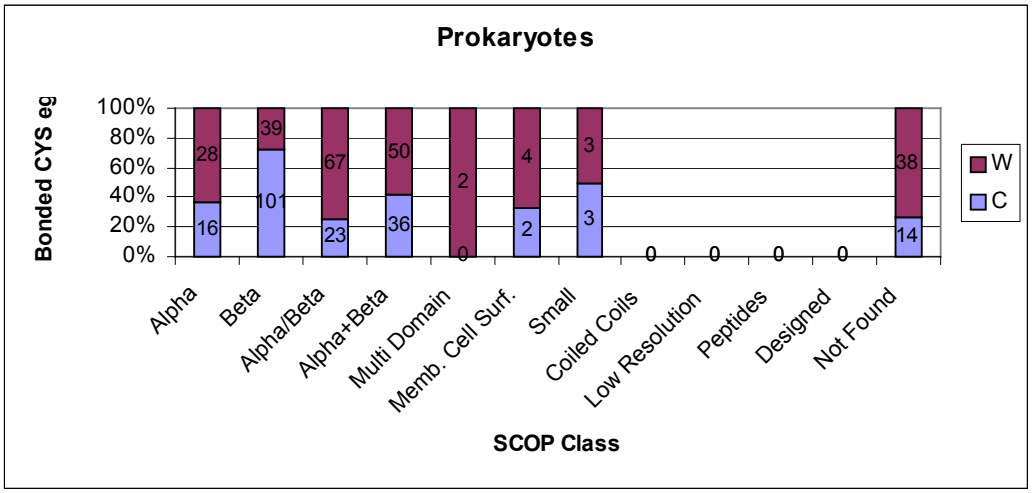


Figure 4.5. Comparison of SCOP classes versus prediction of bonded cysteine examples in (a) Eukaryotes and (b) Prokaryotes. 'C' = Correct Prediction and 'W' = Wrong Prediction. 98% of bonded cysteine examples of 'Small Proteins' in Eukaryotes were found to be correctly predicted.

Importance of disulphide bonds in 'Small Proteins in already known by SCOP definition, that 'Small Proteins' are usually rich in disulphide bonds, which are known to contribute critically to their stability, since they usually lack a strong hydrophobic core.

4.5. Concluding remarks

In this thesis, we have tried to develop a computational method based on machine learning to address a sub-problem of predicting disulphide-bonding states of cysteine residues in protein structures, which will eventually help the field of protein structure prediction to move one step ahead.

We have reached to remarkable accuracies of 94% on cysteine basis for both Eukaryotic and Prokaryotic datasets, and of 90% and 93% on protein basis for Eukaryotic dataset and Prokaryotic dataset respectively. We have obtained these accuracies with a very high correlation value of 0.86 for Eukaryotes and a very decent correlation value of 0.65 for Prokaryotes. These accuracies are best so far ever reached by any existing prediction methods, and thus our prediction methods have outperformed all the previously developed approaches and therefore are more reliable.

Differences in the sequence environment of bonding and free cysteine examples in Eukaryotes and Prokaryotes motivates to do more research for finding if the principal governing factors for disulphide bonding in cysteine residues are different for Eukaryotes and Prokaryotes. Also we found that a set of bonded cysteine (~ 50%) examples in Prokaryotes, which were getting wrongly predicted, have a very low conservation and correlated mutation value in their profiles, and thus have a lower structural importance. This finding is contrary to the general understanding we know about the structural importance of disulphide bonded cysteine residues. Eukaryotes followed it very well and gave a better performance at the basic level of input of profile information.

Prokaryotes (certain types of bacteria and specifically Archaea), which live in very extreme environments (thermophilic, methanogenic, halophilic), are more prone to go under mutations in short time so as to adjust themselves with the environmental conditions and therefore show more diversity in their genomic content. *Malllick et al.* in 2002 has showed that intracellular proteins of archaeal microbes (especially of *Pyrobaculum aerophilum* and *Aeropyrum pernix*) are rich in disulphide bonds, which is contrary to our general understanding that disulphide bonds are mostly found in extracellular environment and rarely in intracellular environment.

The Endoplasmic Reticulum (ER) was long considered to be the only compartment of Eukaryotic cell in which protein folding was accompanied by

enzyme-catalyzed disulphide bond formation. However, in a very recent article by *Riemer et al.* in 2009, has showed that Eukaryotic cells harbor a second oxidizing compartment, the mitochondrial intermembrane space, where disulphide bond formation facilitates protein translocation from the cytosol.

In summery, more study is required to understand the principal governing features of disulphide bond formation in Eukaryotes and Prokaryotes separately, and specifically for Prokaryotes, which because of their extreme habitat show more exceptions in following the already known principals which govern disulphide bond formation.

Bibliography

Alizadeh AA et al., (2000), Distinct type of diffuse large B-cell lymphoma identified by gene expression profiling, *Nature*, Vol. 403, pp. 503–510.

Altschul SF, Gish W, Miller W, Myers EW, and Lipman DJ, (1990), Basic Local Search Alignment Tool, *J. Mol. Biol*., Vol. 215, pp. 403-410.

Baldi P, Soren B, (2001), Bioinformatics: The machine learning approach (2nd edition), *The MIT Press*.

Beeby M, O'Connor BD, Ryttersgaard C, Boutz DR, Perry LJ, and Yeates TO, (2005), The genomics of disulfide bonding and protein stabilization in thermophiles, *PLoS Biol*., Vol. 3(9): e309.

Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN and Bourne PE, (2000), The Protein Data Bank, *Nucleic Acids Research*, Vol. 28 (1) pp. 235-242.

Betz SF, (1993), Disulfide bonds and the stability of globular proteins, *Protein Sci*., Vol. 2, pp. 1551-1558.

Bower JM, and Beeman D, (1995), The Book of Genesis: Exploring realistic neural models with the general neural simulations systems, *Telos/Springer-Verlag, New York.*

Casadio R, Compiani M, Fareiselli P, and Vivarelli F, (1995), Predicting free energy contributions to the conformational stability of folded proteins from the residue sequence with radial basis function networks**,** *Proc. Int. Conf. Intell. Syst. Mol. Biol*., Vol. 3, pp. 81-88.

Ceroni A, Passerini A, Vullo A, and Frasconi P, (2006), DISULFIND: a disulfide bonding state and cysteine connectivity prediction server**,** *Nucleic Acid Research*, Vol. 34, pp. 177-181.

Chen YC, Lin YC, Lin YS, Lin CJ, and Hwang JK, (2004), Prediction of the bonding states of cysteines using the support vector machines based on multiple feature vectors and cysteine state sequences**,** *Proteins*, Vol. 55 (4), pp. 1036-1042.

Creighton T, (1996), Proteins: Structures and Molecular Properties, *W.H. Freeman and Company, New York.*

Fareiselli P, Riccobelli P, and Casadio R, (1999), Role of evolutionary information in predicting the disulfide-bonding state of cysteine in proteins, *PROTEINS: Structure, Function, and Genomics, Vol. 36*, pp. 340-346.

Fariselli P, Finocchiaro G, and Casadio R, (2003), SPEPlip: the detection of signal peptide and lipoprotein cleavage sites, *Bioinfomatics Application Note*, Vol. 19 (18) pp. 2498–2499

Fiser A, Cserzo M, Tudos E, and Simon I, (1992), Different sequence environments of cysteines and half cystines in proteins: application to predict disulfide forming residues, *FEBS Lett*., Vol. 302, pp. 117-120.

Fiser A, and Simon I, (2000), Predicting the oxidation state of cysteines by multiple sequence alignment, *Bioinformatics*, Vol. 16 (3), pp. 251-256.

Forney GD, (1973), The Viterbi Algorithm, *Proceedings of the IEEE*, Vol. 61 (3), pp. 268-278.

Harrison PM, and Sternberg MJE, (1994), Analysis and classification of disulphide connectivity in proteins, *J. Mol. Biol*., Vol. 244, pp. 448–463.

Kabsch W, and Sander C, (1983), Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features, *Biopolymers*, Vol. 22 (12), pp. 2577-637.

Krogh A, and Riis SK, (1999), Neural Computation, *The MIT Press*, Vol. 11, pp. 541–563.

Lesk AM, (2008), Introduction to Bioinformatics (3rd edition), *Oxford University Press.*

Mallick P, Boutz DR, Eisenberg D, and Yeates T, (2002), Genomic evidence that the intracellular proteins of archaeal microbes contain disulphide bonds, *PNAS,* Vol. 99 (15), pp. 9679-9684.

Martelli PL, Fariselli P, Malaguti L, and Casadio R, (2002a), Prediction of the disulfide bonding state of cysteines in proteins with hidden neural networks, *Protein Engineering*, Vol. 15, pp. 951-953.

Martelli PL, Fariselli P, Malaguti L, and Casadio R, (2002b), Prediction of the disulfide-bonding state of cysteines in proteins at 88% accuracy, *Protein Science*, Vol. 11, pp. 2735-2739.

Martelli PL, Fariselli P, Krogh A & Casadio R, (2002c), A sequence-profile-based HMM for prediciting and discriminating β barrel membrane proteins, *Bioinformatics,* Vol. 18 (1), pp. S46-S53.

Mucchielli-Giorgi MH, Hazout S, and Tuffery P, (2002), Predicting the disulfide binding state of cysteines using protein descriptors, *Proteins*, Vol. 46, pp. 243-249.

Muskal SM, Holbrook RS, and Kim SH, (1990), Prediction of the disulfide-bonding state of cysteine in proteins, *Protein Engineering, Design and Selection*, Vol. 3, pp. 667-672

Perou CM, Jeffrey SS, Rijn M, Rees CA, Eisen MB, Ross DT, Pergamenschikov A, Williams CF, Zhu SX, Lee JCF, Lashkari D, Shalon D, Brown PO, and Botstein D, (1999), Distinctive gene expression patterns in human mammary epithelial cells and breast cancers, *Proc. Natl. Acad. Sci. USA*, Vol. 96, pp. 9212–9217.

Pierleoni A, Martelli PL, Fariselli P, and Casadio R, (2006), BaCelLo: a balanced subcellular localization predictor, *Bioinformatics*, Vol. 22, pp. e408-e416.

Pierleoni A, Martelli PL, Fariselli P, and Casadio R, (2007), BaCelLo: a balanced subcellular localization predictor, *Nature Protocols*, DOI: 10.1038/nprot.2007.165

Privalov PL, and Gill SJ, (1988), Stability of protein structure and hydrophobic interaction, *Adv. Protein Chem.*, Vol. 39, pp. 191-234.

Rabiner LR, and Juang BH, An introduction to HMMs, *IEEE ASSP Magazine, vol. 3, 4-16.*

Riemer J, Bulleid N, Herrmann JM, (2009), Disulphide formation in the ER and mitochondria: Two solution to a comman process, *Science*, Vol. 324, pp. 284-1287.

Rost B, Sander C, (2004), Combining evolutionary information and neural networks to predict protein secondary structure, *Proteins*, Vol.19 (1), pp. 55 – 72.

Russell S, and Norvig P, (2009), Artificial Intelligence. *Printice Hall series*.

Sevier CS, and Kaiser CA, (2002), Formation and transfer of disulphide bonds in living cells, *Nature Reviews Molecular and Cellular Biology*, Vol. 3, pp 836-847.

Song JN, Wang ML, Li WJ, and Xu WB, (2004), Prediction of disulphide bonding state of cysteines in proteins based on dipeptide compositon, *Biochem. Biophys. Res. Commun.*, Vol. 318, pp. 142-147.

Tarca AL, Carey VJ, Chen XW, Romero R, Draˇghici S, (2007), Machine learning and its applications to biology, *PLoS Comput. Biol.*, Vol. 3 (6), pp. 954-963.

Vassura M, Margara L, Di Lena P, Medri F, Fariselli P, and Casadio R, (2008), Reconstruction of 3D Structures From Protein Contact Maps, Reconstruction of 3D Structures From Protein Contact Maps, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol. 5 (3).

Vieille C, and Zeikus G.J, (2001), Hyperthermophilic enzymes: Sources, uses, and molecular mechanisms for thermostability, *Microbiol. Mol. Biol.*, Vol. 65, pp. 1-43.

Viterbi AJ, (1967), Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, *IEEE Transactions on Information Theory*, Vol.13 (2), pp. 260-269.

Wedemeyer WJ, Welkler E, Narayan M, and Scheraga HA, (2000), Disulfide Bonds and Protein Folding, *Biochemistry*, Vol. 39, pp. 4207–4216.