PROTEIN FOLD RECOGNITION USING

ADABOOST LEARNING STRATEGY

Yijing Su

Submitted to the faculty of the School of Informatics
in partial fulfillment of the requirements
for the degree of
Master of Science in Bioinformatics,
Indiana University

December, 2007

Accepted by the Faculty of Indiana University,
in partial fulfillment of the requirements for the degree of Master of Science
in Bioinformatics

Master's Thesis
Committee

_____
Jeffrey Huang, Ph.D., Chair

_____
Yuni Xia, Ph.D.

_____
Jiang Yu Zheng, Ph.D.

Dedicated to my husband, my son, and my parents.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

A sincere wish of gratitude for all who have supported me through these years of study.

ABSTRACT

Yijing Su

PROTEIN FOLD RECOGNITION USING ADABOOST STRATEGY

Protein structure prediction is one of the most important and difficult problems in computational molecular biology. Unlike sequence-only comparison, protein fold recognition based on machine learning algorithms attempts to detect similarities between protein structures which might not be accompanied with any significant sequence similarity. It takes advantage of the information from structural and physic properties beyond sequence information. In this thesis, we present a novel classifier on protein fold recognition, using AdaBoost algorithm that hybrids to k Nearest Neighbor classifier. The experiment framework consists of two tasks: (i) carry out cross validation within the training dataset, and (ii) test on unseen validation dataset, in which 90% of the proteins have less than 25% sequence identity in training samples. Our result yields 64.7% successful rate in classifying independent validation dataset into 27 types of protein folds. Our experiments on the task of protein folding recognition prove the merit of this approach, as it shows that AdaBoost strategy coupling with weak learning classifiers lead to improved and robust performance of 64.7% accuracy versus 61.2% accuracy in published literatures using identical sample sets, feature representation, and class labels.

[READ THIS ABSTRACT ABOVE.]

CHAPTER ONE: INTRODUCTION

Proteins are an important class of biological macromolecules present in all biological organisms. Proteins have different levels of structural organization and generally fold into one or more specific spatial conformations, driven by a number of noncovalent interactions such as hydrogen bonding, ionic interactions, Van der Waals forces, and hydrophobic packing. Most proteins can carry out their biological functions only when folding has been completed, because three-dimensional shape of the proteins in the native state is critical to their function (Malacinski, 2003).

In order to understand the functions of proteins at a molecular level, it is necessary to determine the three dimensional structure of proteins. The two major laboratory methods available for studying protein folding structure, X-ray crystallography and Nuclear Magnetic Resonance (NMR) spectroscopy, are time consuming and expensive (Baldi and Brunak, 2001). The current proteomic study shows that the number of known protein sequence discovered from wet bench grows exponentially every year while the progress of determining protein structure is still far behind sequence finding. Generally the ratio of the number of known amino acid sequences to the number of validated three dimensional protein structures is about 100 to 1, while this gap continuously widen every year (Okun, 2004). Toward this end, extracting structural information automatically from sequence databases is critically needed and predicting structure relying on computational algorithms are becoming more and more important. Protein structure prediction from the amino acid sequence information have tremendous impact in all of biotechnology and drug design, and it is a main stream in bioinformatic

proteome research (Rost and Sander,1994). A wide range of machine learning approaches have been extensively applied for the prediction of protein structural classes and folds. Among these various machine learning tools, neural networks (NNs) have been quite successful for such prediction (Ding and Dubchak, 2001). Multi-layer perceptron (MLP) network (Bologna and Appel, 2002) and radial basis function (RBF) network (Chung *et al.*, 2003) have been used broadly to protein fold determination. In recent years, along with the achievement of support vector machines (SVMs) made in pattern classification including bioinformatics applications, SVMs have also been used in protein fold determination (Ding and Dubchak, 2001; Chung *et al.*, 2003).

Much progress has been made by many research groups that are interested in such kind of tasks. When categorizing protein structure into 4 major classes - all α, all β, α/β, and $\alpha+\beta$, a higher than 70% prediction accuracy can be achieved by various classification methods mentioned above, using sets of vector presentation such as simple frequency feature as well as general features extracted from a collection of protein sequences (Dubchak *et. al*., 1999; Chung *et. al*., 2003). However, categorizing proteins into these 4 structure classes is generally the first step in summarizing protein folding. When we are further interested in knowing the details of folding structure beyond these 4 structure classes, such as globin-like(which belongs to all α), immunoglobulin-like (which belongs to all β), (TIM)-barrel (which belongs to α/β), prediction accuracy often degrades rapidly with respect to the increasing number of classes. It is a fact that it is more challenging to deal with multiple class label classification when the number of classes is large (Chung *et. al*., 2003).

The objective of my thesis work is to investigate and develop a computational frame work which for *m* class classification while *m* is a large number. To make fair comparison among state-of- art computational algorithms, we focus our effort only on those research groups which have performed their methods on same working datasets, which contain 27 classes. The existing literature report an overall successful recognition rate is about 61.2% so far (Bologna and Appel, 2002)

Our experiments consist of two innovative pattern classification methods to predict the fold pattern from query protein sequences: (i) ensemble classifier using hybrid multi-layer back-propagation neural networks, and (ii) AdaBoost algorithm hybrids to k nearest neighbor classifier. The performance is compared to the baseline algorithm, k nearest neighbor algorithm, and further challenges the outcomes from the existing research groups.

The organization of this thesis is as follows: Chapter Two gives an introduction of protein structure and its relationship with function, as well as protein structure prediction using computational techniques; Chapter Three is the literature review, explaining the way of extracting features from amino acid sequences and briefly listing state of the art for protein structure prediction with the emphasis on the research groups involving the same working datasets; Chapter Four illustrates the methodology used in this thesis; Chapter Five describes the experiments setup, system architecture and data preparation, and shows the performance; Chapter Six compares this novel classifier with the benchmarks and makes some discussion on the contribution of AdaBoost algorithm in the tasks of protein fold recognition.

CHAPTER TWO: BACKGROUND

This chapter gives an introduction of the topic in this thesis and illustrates the importance of this topic, by providing some background knowledge and information from the biological side to computational side. The organization is as following: (i) protein structure and its relationship with function, giving an introduction of protein structure and explaining why put efforts on studying protein structure; (ii) protein structure classification systems, introducing the existing protein structure databases; (iii) protein structure determination, briefly describing two experimental techniques used in protein structure determination; and (iv) computational prediction of protein structure, illustrating three major theoretical methods for predicting the structure of proteins using computational techniques.

Protein Structure and its Relationship with Function

Proteins are an important class of biological macromolecules present in all biological organisms, made up of such elements as carbon, hydrogen, nitrogen, oxygen, and sulfur. Proteins and peptides form the very basis of life, by regulating a variety of activities in all known organisms, and are generally responsible for regulating the cellular machinery and consequently, the phenotype of an organism (Malacinski, 2003). Proteins have different levels of structural organization including:

(i) Primary structure refers to the amino acid sequence of the peptide chains, which can be looked as the "linear" sequence of amino acids.

(ii) Secondary structure is the "local" ordered structure in proteins and is mainly formed through hydrogen bonds between backbone atoms. The most common secondary structure elements in proteins are the α-helix and the β-sheet. Secondary structure is defined as the local conformation of its backbone.

(iii) Tertiary structure is the "global" folding of a single polypeptide chain, describing the packing of α-helices, β-sheets and random coils with respect to each other on the level of one whole polypeptide chain. A major driving force in determining the tertiary structure of globular proteins is the hydrophobic effect. The polypeptide chain folds such that the side chains of the nonpolar amino acids are "hidden" within the structure and the side chains of the polar residues are exposed on the outer surface. Tertiary structure of a protein is formed when the attractions of side chains and those of the secondary structure combine and cause the amino acid chain to form a distinct and unique 3-dimensional structure. It is this unique structure that gives a protein is specific function.

(iv) Quaternary structure involves the association of two or more polypeptide chains into a multi-subunit structure. Quaternary structure is the stable association of multiple polypeptide chains resulting in an active unit. Not all proteins exhibit quaternary structure. It only exists when there is more than one polypeptide chain present in a complex protein. Then quaternary structure describes the spatial organization of the chains.

Most proteins can carry out their biological functions only when folding has been completed, because three-dimensional shape of the proteins in the native state is critical to their function. The secondary structures, the most common of which are α-helix and β-

sheet, are formed by a small number of amino acids that are close together, which then, in turn, interact, fold and coil to produce the tertiary structure that contains its functional regions (Malacinski, 2003). In order to understand the functions of proteins at a molecular level, it is necessary to determine the three dimensional structure of proteins.

Protein Structure Classification Systems

There are several databases existing to identify groups of similarly folded proteins. SCOP, CATH, and FSSP are the largest ones. CATH is a semi-automatic, hierarchical classification of protein domains, which clusters proteins at four major levels: Class (C), Architecture (A), Topology (T), and Homologous superfamily (H). SCOP (Structured Classification of Proteins) is a largely manual classification of protein structural domains based on similarities of their amino acid sequences and three-dimensional structures, providing a detailed and comprehensive description of the structural and evolutionary relationships among all proteins whose structures are known. SCOP utilizes four levels of hierarchic structural classification: Class, Fold, Superfamily, and Family. This classification is more significantly based on the human expertise than semi-automatic CATH, its chief rival. It is usually accepted that SCOP provides a better justified classification (Baldi and Brunak, 2001). Human expertise is needed to decide whether certain proteins are evolutionary related and therefore should be assigned to the same superfamily, or their similarity is a result of structural constraints and therefore they belong to the same fold. FSSP is both known as Fold classification based on Structure-Structure alignment of Proteins and Families of Structurally Similar Proteins. FSSP is

purely automatically generated (including regular automatic updates) but offers no classification, allowing the user to draw their own conclusion as to the significance of structural relationships based on the pairwise comparisons of individual protein structures. Although using different methods and there are some differences in these databases, the classification of the majority of proteins which have been classified is consistent in general (Baldi and Brunak, 2001).

Protein Structure Determination

As of today, there are two experimental techniques, which are namely, X-ray crystallography and Nuclear Magnetic Resonance (NMR) spectroscopy, available to determine the 3D structure of proteins. Around 90% of the protein structures in the Protein Data Bank are determined by X-ray crystallography. This method allows one to measure the 3D density distribution of electrons in the protein (in the crystallized state) and thereby infer the 3D coordinates of all the atoms to be determined to a certain resolution. Roughly 9% of the known protein structures are obtained by NMR techniques (Malacinski, 2003). However, these two laboratory methods are time consuming and expensive. And the current proteomic study shows that the number of known protein sequence discovered from wet bench grows exponentially every year while the progress of determining protein structures is still far behind sequence finding. Generally the ratio of the number of known amino acid sequences to the number of validated 3D structures is about 100 to 1, while this gap continuously widen every year (Okun, 2004). Toward this

end, the practical role of protein structure prediction using computational methods is now more necessary and more important than ever.

Computational Prediction of Protein Structure

Due to the fact that the structure of a protein gives much more insight in the function of the protein than its sequence, as well as the large gap between the number of known amino acid sequences and the number of known protein structures, a wide range of methods for the computational prediction of protein structure from its sequence have been proposed. There are certainly a number of factors make it a very difficult task, including the number of possible structures that proteins may possess is extremely large, the physical basis of protein structural stability is not fully understood, the tertiary structure of a native protein may not be readily formed without the aid of transacting factors, one particular sequence may be able to assume multiple conformations depending on its environment, and so on (Malacinski, 2003).

In spite of the above hindrances, much progress has been made by the many research groups that are interested in the task. Prediction of structures for small proteins is now a perfectly realistic goal. There are three major theoretical methods for predicting the structure of proteins: *Ab initio* prediction, comparative modeling, and fold recognition.

(i) *Ab initio* protein modeling method is a mixture of science and engineering, seeking to build 3D protein models from scratch, i.e., based on physical principles rather than directly on previously solved structures. The science portion is in understanding how the 3D structure of proteins is attained, while the engineering one is in deducing the

3D structure by a given sequence. This modeling method tends to require vast computational resources, and have thus only been carried out for tiny proteins. To attempt to predict protein structure *de novo* for larger proteins, we will need better algorithms and larger computational resources like those afforded by either powerful supercomputers or distributed computing (Pevzner, 2000).

(ii) Comparative protein modeling, also known as homology modeling, is based on the reasonable assumption that two homologous proteins will share very similar structures. Since it is widely accepted that a protein's fold is more evolutionarily conserved than its amino acid sequence; this method uses previously solved structures as starting points or templates to predict the structure of the target sequence. Unsurprisingly, homology modeling is most accurate when the target and template have similar sequences (Malacinski, 2003). But, it is quite often that the query protein does not have any structure-known homologous protein in the existing databases.

(iii) Protein fold recognition or threading scans the amino acid sequence of an unknown structure against a database of solved structures, producing a list of scores. The scores are then ranked and the fold with the best score is assumed to be the one adopted by the sequence (Malacinski, 2003). Fold recognition methods are widely used and effective because it is believed that there are a strictly limited number of different protein folds in nature, mostly as a result of evolution but also due to constraints imposed by the basic physics and chemistry of polypeptide chains (Dubchak et. al., 1999).

Based on the statistics of SCOP, currently there are 971 different protein folds known (from SCOP database statistics for 1.71 release (October 2006)), and new folds are still being discovered every year thanks in part to the ongoing structural genomics

projects. By the assumption that the number of protein folds is restricted, predicting the three-dimensional structure of a protein may be preliminary converted to a particular classification problem. This thesis follows this approach and does some research based on it.

When studying the protein structure prediction problems, a primitive way to categorize protein structure into four major classes is generally used. This is systematically based upon α-helix and β-sheet as well as the general topological properties of the mixture of both. As described in chapter two, α-helices and β-sheets are two well-defined secondary structural units abundantly existing in proteins. Different topology may constitute a specific type of fold (Malacinski, 2003). These four major classes are:

(i) all α: the secondary structure is almost exclusively α –helices;

(ii) all β: the secondary structure is composed almost exclusively of β –sheets;

(iii) α/β: helices and sheets are arranged in the sequence of β-α-β units and the β -sheet strands are in parallel;

(iv) $\alpha+\beta$: helices and sheets tend to be spatially separated in different parts of the protein and lack of β-α-β supersecondary structure.

Beyond the 4 major structural classes, there is deeper level classification of protein folds. Table 1 lists the number of protein folds based on SCOP statistics for 1.71 releases (October 2006). Accordingly, categorizing protein structure into these 4 major classes is generally the first step in summarizing protein folding, and we are further interested in knowing the details of folding structure beyond these 4 major classes, such

as globin-like(which belongs to all α), immunoglobulin-like (which belongs to all β), (TIM)-barrel (which belongs to α/β), etc.

Table 1. Protein structure classes with the number of protein folds based on SCOP statistics for 1.71 releases (October 2006).

Class	Number of folds
All α proteins (essentially formed by α –helices)	226
All β proteins (essentially formed by β –sheets)	149
α/β proteins (β-α-β units with parallel β –sheets)	134
α+β proteins (lack of β-α-β, segregated α *and* β regions)	286
Others (membrane and cell surface proteins, small proteins, *etc.)*	176
Total	971

CHAPTER THREE: LITERATURE REVIEW

The section of the literature review will go over some progress made on protein structure prediction using various computational methods. This chapter explains the way of extracting features from amino acid sequences and briefly lists the various computational algorithms for protein structure prediction with the emphasis on the research groups involving the same working datasets.

To systematically illustrate the machine learning methods of protein structure prediction and to make fair comparison among state-of- the-art computational algorithms, this thesis focuses the effort only on those research groups which have performed their efforts on the same working datasets using the same methods to extract features from amino acid sequences, which will be discussed below.

The reason why this thesis chooses the working datasets is the challenge of dealing with multiple class label classification problem when the number of classes is large. In a broad structural classification, which contains four major classes - all α, all β, α/β, and $\alpha+\beta$ (as described in chapter two), a higher than 70% prediction accuracy can be achieved by various classification methods, using set of vector presentation such as simple frequency feature as well as general features extracted from a collection of protein sequences (Dubchak et. al., 1999; Chung et. al., 2003). However, when the classification goes into a deeper level, protein fold recognition, the prediction accuracy often degrades rapidly with respect to the increasing number of classes. It is fact that it is more challenging to deal with multiple class label classification problem when the number of classes is large.

Feature Extraction

The working datasets, used for training and testing by those research groups (which will be discussed in the next section), are taken from Ding and Dubchak (2001). The proteins in the training and testing datasets are classified into the 27 fold types according to the SCOP database (Andreeva *et al*, *2004;* Murzin *et al*, 1995). Among the above 27 fold types, there are 6 types belong to all α structural class, 9 types to all β class, 9 types to α/β class, and 3 types to $\alpha+\beta$ class. Therefore, the classification of 27 folds is one level deeper and more challenge than that of 4 major structural classes. To design experiments utilizing machine learning approach and applying learning-from-example strategy, six feature subsets are extracted from protein sequences based on the following manners (Dubchak *et al*, 1999; Dubchak *et al*, 1995). Percentage composition of the twenty amino acids forms the first feature subset consisting of 20 features, named as *amino acids composition*. The other five subsets of feature vectors are named *predicted secondary structure*, *hydrophobicity*, *normalized van der Waals volume*, *polarity*, and *polarizability*. Each of these five feature subsets contains 21 dimension feature vectors. Thus, a feature vector concatenate six feature subsets and produces 125 feature values in total ($20 + 21 \times 5$).

In order to understand how to extract these feature vectors from amino acid sequences, here we will briefly introduce the approach used in Dubchak *et al* (1995, 1999), which used a combination of local and global information about amino acid sequences and constructed in two steps.

In the first step, twenty amino acids were divided into three groups for each subset representing the main clusters of the amino acid index of Tomii and Kanehisa

(1996**).** Thus, for each subset, every amino acid was replaced by the index 1, 2, or 3 according to one of the three groups to which it belonged. For the subsets *predicted secondary structure*, the index 1, 2, and 3 correspond to the helix, strand, and coil, respectively. For the other four attributes, those of *hydrophobicity*, *normalized van der Waals volume*, *polarity*, and *polarizability*, the 20 amino acids were divided into three groups according to the magnitudes of their numerical values. The ranges of these numerical values and the amino acids belonging to each group are shown in Table 2.

Table 2. Amino acid attributes and the division of the amino acids into 3 groups for each attribute (source from Dubchak *et al*, 1999)

Property	**Group 1**	**Group 2**	**Group 3**
Hydrophobicity	Polar R, K, E, D, Q, N	Neutral G, A, S, T, P,H,Y	Hydrophobic C, V, L, I, M, F,W
Normalized van der Waals volume	0–2.78 G, A, S, C, T, P, D	2.95–4.0 N, V, E, Q, I, L	4.43–8.08 M, H, K, F, R,Y,W
Polarity	4.9–6.2 L, I, F,W, C, M, V,Y	8.0–9.2 P, A, T, G, S	10.4–13.0 H, Q, R, K, N, E, D
Polarizability	0–0.108 G, A, S, D, T	0.128–0.186 C, P, N, V, E, Q, I, L	0.219–0.409 K, M, H, F, R,Y,W

In the second step, three descriptors were calculated for a given subset: (i) Composition (C), to describe the global percent composition of each of the three groups in a protein; (ii) Transition (T), to describe the percent frequencies with which the attribute changes its index along the entire length of the protein; and (iii) Distribution (D), to describe the distribution pattern of the attribute along the sequence (Dubchak *et al*, 1995, 1999).

Let us consider the *hydrophobicity* attribute as an example (see table 3). The model sequence is *LTKDEYERHNSYTCEATHKTSTSP*. Based on the partition rules discussed above, all amino acids are divided into three groups, polar, neutral, and hydrophobic, it can be transferred into 32111211212223122212222 2.

Table 3. Model sequence consisting of 3 types of residues (Group 1, 2 and 3) to describe the feature vectors of *hydrophobicity*

Numbering	1				5					10					15					20					
Sequence	L	T	K	D	E	Y	E	R	H	N	S	Y	T	C	E	A	T	H	K	T	S	T	S	P
Group	3	2	1	1	1	2	1	1	2	1	2	2	2	3	1	2	2	2	1	2	2	2	2	2
Group 1			*	*	*		*	*		*					*				*					
Group 2		*				*			*		*	*	*			*	*	*		*	*	*	*	*
Group 3	*													*										
1-2 transitions			*			*	*		*	*	*					*			*	*				
1-3 transitions															*									
2-3 transitions		*												*										

The model sequence includes 8 Group 1 residues ($n_1 = 8$), 14 Group 2 residues ($n_2 = 14$) and 2 Group 3 residues ($n_3 = 2$). The percent compositions are calculated as follows: $n_1 / (n_1 + n_2 + n_3) = 33.3\%$ for Group1, $n_2 / (n_1 + n_2 + n_3) = 58.3\%$ for Group2, and $n_3 / (n_1 + n_2 + n_3) = 8.3\%$ for Group3. These three numbers represent the first descriptor, C, the global percent compositions of polar, neutral, and hydrophobic residues in the protein. The second descriptor, T, also consists of the three numbers -- the percent frequency with which: (i) a polar residue is followed by a neutral residue or a neutral residue by a polar residue, in which case, there are 9 transitions of this type, that is $(9 / 23) \times 100\% = 39.1\%$; (ii) a polar residue is followed by a hydrophobic residue or a hydrophobic residue by a polar residue, in which case, there are 1 transitions of this type, that is $(1 / 23) \times 100\% = 4.3\%$; and (iii) a neutral residue is followed by a hydrophobic residue or a hydrophobic residue by a neutral residue, in which case, there are 2 transitions of this type, that is $(2 / 23) \times 100\% = 8.7\%$. The third descriptor, D, consists of the five numbers for each of the three groups: the fractions of the entire sequence, where the first residue of a given group is located, and where the 25%, 50%, 75%, and 100% of those are contained. In this example, the first residue of Group 1 coincides with the first 3 of the chain, so the first number of D descriptor equals $(3 / 24) \times 100\% = 12.5\%$. 25% of all Group 1 residues ($25\% \times 8 = 2$ residues) are contained within the first 4 residues of the protein chain, so the second number equals $(4 / 24) \times 100\% = 16.7\%$. Similarly, 50% of Group 1 residues ($50\% \times 8 = 4$ residues) are within the first 7 residues of the chain; thus, the third number is $(7 / 24) \times 100\% = 29.2\%$. The fourth and the fifth numbers of the distribution descriptor are 41.7% and 79.2%, respectively. Analogous numbers for Group 2 are 8.3%, 37.5%, 66.7%, 83.3%, and 100%, respectively. Analogous numbers

for Group 3 are 0.0, 0.0, 4.2%, 4.2%, and 58.3%, respectively. Thus, the complete parameter vector contains 3 (C) + 3 (T) + 5 × 3 (D) = 21 scalar components. They are 33.3, 58.3, 8.3, 39.1, 4.3, 8.7, 12.5, 16.7, 29.2, 41.7, 79.2, 8.3, 37.5, 66.7, 83.3, 100.0, 0.0, 0.0, 4.2, 4.2, and 58.3.

Within the feature extraction method described above, feature vectors can be easily calculated from new sequences, and fold prediction by different machine-learning techniques can be performed rapidly and automatically.

State-of-the-Art

When dealing with the classification within four classes - all α, all β, α/β, and α+β, a higher than 70% prediction accuracy can be achieved by various classification methods, using the sets of feature vectors extracted from protein sequences by the methods illustrated in previous section (Dubchak et. al., 1999; Chung et. al., 2003). However, when the classification goes into a deeper level, protein fold recognition, the prediction accuracy often degrades rapidly with respect to the increasing number of classes.

All the approaches, briefly analyzed below, use the working datasets, which contains 27 classes and whose feature vectors are extracted from protein sequences by the methods described above. Unless otherwise stated, a 125 dimensional feature vector is assumed for each protein fold.

Ding and Dubchak (2001) employed one-versus-others, unique one-versus-others and all-versus-all methods to utilize many two-class classifiers (for example, 27 two-way

classifiers for one-versus-others, $27 \times (27 - 1) / 2 = 351$ two-way classifiers for all-versus-all methods) as the building blocks of this multi-class classification problem. In their work, support vector machines (SVMs) and three-layer feed-forward neural networks (NNs) were used.

Bologna and Appel (2002) used a 131 dimensional feature vector and an ensemble of four-layer Discretized Interpretable Multi-Layer Perceptrons (DIMLP). In their working datasets, the protein length for each protein fold is reported in each of the six feature subsets in addition, thus the feature vector concatenate produced 131 feature values in total ($21 + 22 \times 5$). In DIMLP, each network learns all protein folds simultaneously, which is in contrast to Ding and Dubchak's work (2001). Bagging and arcing algorithms were employed to combine the outputs of DIMLPs individually.

Chung *et al.* (2003) designed a hierarchical two-level classification and selected Neural Networks (NNs) and SVMs as base classifiers. At first level of classification, a protein to be recognized was assigned to one of the four classes (all α, all β, (α+β), or α/β). At second level, it was classified as one of the 27 folds. This level employed the outputs of the first level, which meant they were not trained to predict all folds, but only those belonging to a certain structural class. In contrary to Ding and Dubchak's work (2001), each NN or SVM is a multi-class classifier so that the number of classifiers is greatly reduced (actually, it is equal to five: one classifier for class recognition and the four for fold recognition). The common NN models including Multi-Layer Perceptron (MLP), Radial Basis Function Network (RBFN), and General Regression Neural Network (GRNN) with a single hidden layer were used.

Okun (2004) explored new techniques by modifying the standard K-nearest neighbor algorithm intending to improve the classification performance of the conventional KNN to a level of SVM. Unlike the SVM, which builds a nonlinear decision surface, separating different classes of the data, in a high (often infinite) dimensional feature space, K-Local Hyperplane Distance Nearest Neighbor algorithm (HKNN) tries to find this surface directly in input space. It was assumed that each class was locally linear in such a high dimensional space. The idea of HKNN was to fantasize the missing points in the manifold, which introducing artifacts in the decision surface generated by the conventional KNN, thus negatively affecting the generalization ability of KNN, based on a local linear approximation of the manifold of each class.

CHAPTER FOUR: METHODOLOGY

In this thesis, there are two independent machine learning methods are employed and explored for the tasks of protein fold classification. One is the ensemble classifier using hybrid multi-layer neural networks, in which the back-propagation neural networks (BP-NN) are used and data fusion approach is employed to hybrid the intelligence classifiers. The other is AdaBoost algorithm hybrids to k nearest neighbor (KNN) classifier, in which k nearest neighbor algorithm is brought into play as the baseline algorithm to generate weak classifiers and AdaBoost algorithm is for improving the accuracy of k nearest neighbor algorithm. Here we will provide a brief description of these baseline classifiers and ensemble algorithms individually.

Neural Networks

Neural networks (NNs) have been trained to perform complex functions in various fields of application including pattern recognition, identification, classification, speech, vision and control systems (Haykin, 1998). Today neural networks can be trained to solve problems that are difficult for conventional computers or human beings and have also been widely used for protein fold determination (Malacinski, 2003). In this thesis, we use back-propagation neural networks, which are a popular type of network that can be trained to recognize different patterns. The back-propagation networks consist of several layers of neurons of which first one is the input layer and the last one is the output layer, remaining layers are called hidden layers. The number of neurons in the input layer depends on the number of possible inputs we have, while the number of

neurons in the output layer depends on the number of desired outputs. The number of hidden layers and how many neurons in each hidden layer cannot be well defined in advance, and could change depending on the network configuration and the types of data. By offering such a degree of data compression or expansion, neural networks are good choices in the hybrid systems, in which logical processing layer is generally compact and fuse data from different feature modalities and cognitive modes.

Data Fusion

Data fusion is the process of putting together information obtained from many heterogeneous sensors, on many platforms, into a single composite picture of the environment. We employ it here to hybrid intelligent classifiers by constructing a multi-layer pattern classification system for the purpose of fusing distinct modalities of folding feature vectors. The concept of reductionism is a common practice in the development of intelligent systems - to design solutions to complex problems through a stepwise decomposition of the task into successive modules (Kuncheva *et al*, 2001). Typically, in hybrid systems, reflexive tasks are assigned to the connectionist subsystem and deliberative tasks to the second level of classifier. The hybrid approach for classification involves specific levels of knowledge where the hierarchy is defined in terms of concept granularity and specific interfaces (Oza *et al*, 2005). As one moves upward in the hierarchical structure, we witness a corresponding degree of data compression so more powerful ('reasoning') methods can be employed on reduced amounts of data. Connectionism can handle the whole range of sensory inputs and their variability

('noise'). Its distributed nature provides for fault tolerance to missing and incomplete data. The output of such modules, known to have a well-defined maximum likelihood (ML) probabilistic meaning, can be thus combined across ensemble of such networks. Symbolic methods are compact and can fuse data from different sensory modalities and cognitive modes. As a consequence one can make sense of the sensory input and interpret ('explain') it using meaningful coding units (Oza *et al*, 2005).

The hybrid classifier architecture for protein folding prediction tasks pursued in this thesis consists of two layers of back-propagation (BP) networks. The system architecture is shown in Figure 1. This algorithm uses ensemble neural network computation for protein folding classification. The hybrid intelligent classifiers consist of a set of ensemble networks using back-propagation and the network outputs are summarized by another back-propagation neural network. At first layer, several BP-NNs are used individually. A single Back-propagation neural network at the second layer of system architecture implements the fusing stage using the first layer outputs as input vectors. The CV ensembles implements active learning schemes leading to increased ambiguity by employing different topologies for the networks themselves and training the networks on different data sets corresponding to variations of the original data.

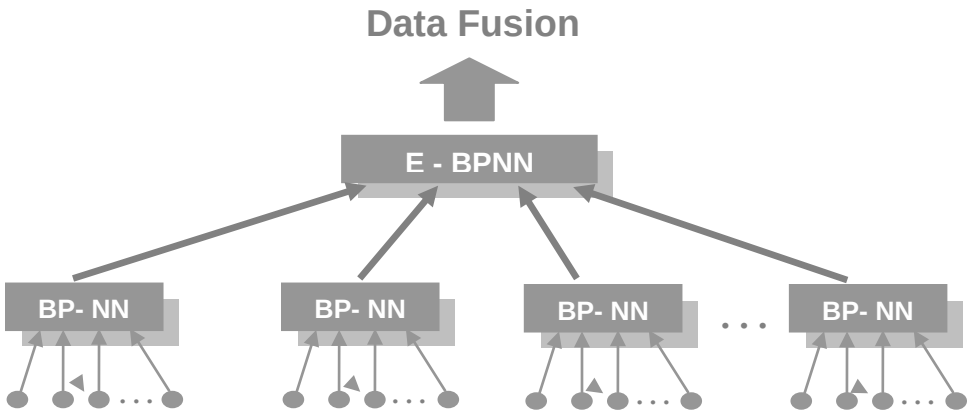

Figure 1. The architecture of the hybrid classifier system

AdaBoost Algorithm

AdaBoost, also named as “adaptive boosting" algorithm, introduced by Freund and Schapire (1996, 1997), with the goal of improving the accuracy of any given learning algorithm. It works by incrementally adding one hypothesis at a time to an ensemble classifier. In AdaBoost, each training sample receives a weight, which is initialized to be uniform, to determine its probability of being re-selected for training classifier in next iteration. If a training pattern is accurately classified, then its chance of being used again in a subsequent component classifier is reduced. Conversely, if the pattern is misclassified, then its chance of being used again is raised. AdaBoost.M1 is the most straightforward extension of AdaBoost used for multi-class classifier (Dietterich, 2000).

We let the feature vector of the samples and their labels in D be denoted by x^i and y_i, respectively, let C_k be the classifier on iteration k, and let $W_k(i)$ be the kth discrete

distribution over all these training samples. The AdaBoost.M1 pseudocode (Dietterich, 2000), is as following:

---

1. Begin initialize $D = \{x^1, y_1, x^2, y_2, ..., x^n, y_n\}$, $k_{\max}$, $W_1(i) = 1/n$, $i = 1, ..., n$

2 $k \leftarrow 0$

3 Do $k \leftarrow k+1$

4 Train weak learner C_k using D sampled according to distribution $W_k(i)$

5 $E_k \leftarrow$ Training error of C_k measured on D using $W_k(i)$

6 $\alpha_k \leftarrow 1/2 \ln[(1 - E_k)/E_k]$

7 $$W_{k+1}(i) \leftarrow \frac{W_k(i)}{Z_k} \times \begin{cases} e^{-\alpha_k} & \text{if } h_k(x^i) = y_i \quad * \\ e^{\alpha_k} & \text{if } h_k(x^i) \neq y_i \quad ** \end{cases}$$

8 Until $k = k_{max}$

9 Return C_k and α_k for $k = 1$ to k_{max} (ensemble of classifiers with weights)

10 End

* Correctly classified

** Incorrectly classified

In line 5, E_k , the error for classifier C_k, is determined with respect to the distribution $W_k(i)$ over D on which it was trained. In line 7, Z_k is chosen to normalize W_{k+1}; while $h_k(x^i)$ is the category label (+1 or -1) given to pattern x^i by weak classifier C_k. The final classification of a testing sample vector x is based on a discriminant function that is the weighted sum of the outputs given by the component classifiers $H = \arg\max_{y \in Y} \sum_{k=1}^{K} \alpha_k h_k$.

Figure 2 shows the system architecture of AdaBoost hybrid weak classifiers.

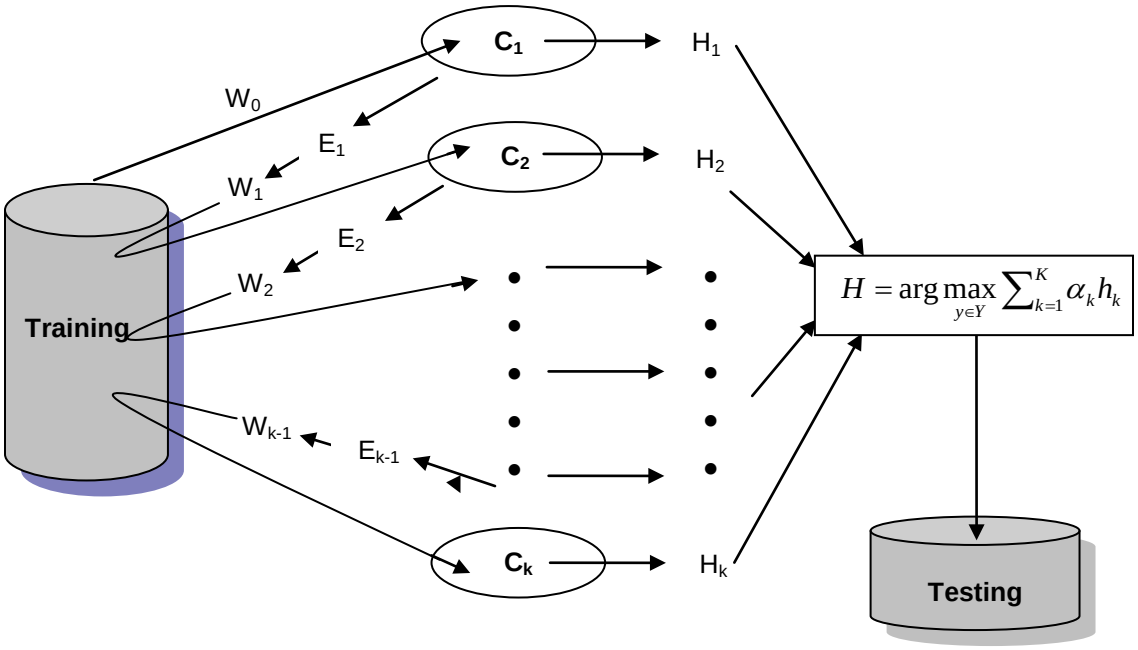

Figure 2. The architecture of AdaBoost hybrid weak classifiers system

K Nearest Neighbor Algorithm

K Nearest Neighbor algorithm (KNN) was first introduced by the researchers Fix and Hodges (1951) for classifying objects based on closest training examples in the feature space. The training examples are mapped into multidimensional feature space, thus, the space is partitioned into regions by class labels of the training samples. A point in the space is assigned to the class *c* if it is the most frequent class label among the k nearest training samples, where k is the number of neighbors. Usually *Euclidean distance* is used. The Euclidean distance between two points $P = (p_1, p_2, \ldots, p_n)$ and $Q = (q_1, q_2, \ldots, q_n)$, in Euclidean *n*-space, is defined as:

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

The best choice of k depends upon the data; generally, larger values of k reduce the effect of noise on the classification, but make boundaries between classes less distinct.

In this thesis, we use AdaBoost Algorithm to hybrid on k nearest neighbor classifiers. As discussed in previous section, in AdaBoost, each training sample receives a weight which determines its probability of being re-selected for training classifier in next iteration. To integrate these two algorithms, we employ the weight for each training sample in each of iteration in AdaBoost to k nearest neighbor algorithm. We let *n* be the number of samples in *D*, whose weight, feature vector, labels are denoted by w_i, x^i, and y_i, where $y_i \in \{1,...,k\}$, respectively. We let *NN* be the number of the nearest neighbor and let *NS* be the neighbor set. The pseudocode of modified KNN is as following:

1. Begin initialize $D = \{x^1, y_1, x^2, y_2, ..., x^n, y_n\}$, $w_i = w_i \times n$, $i = 1, ..., n$

2. For each x^i

3. Calculate the distances $d_{i,j}$ between X^i and X^j, $j = 1, ..., i-1, i+1, ..., n$

4. Sort $d_{i,j}$ in increasing order

5. According the sorted distances, put the first *m* records into the neighbor set *NS* so that $\sum_{X^j \in NS} w_j < NN$

6. According to the labels y_j, separate the neighbor set *NS* into *k* subsets $S_k (k = 1...K)$, and calculate the weights W_{S_k} of each subsets S_k

$$W_{S_k} = \sum_{X^i \in S_k} w_i$$

7. Output the hypothesis *h*: $X \rightarrow Y$

Assign x^i class label from S_k where $\max_{k=1...K} W_{S_k}$ is found

8. End

CHAPTER FIVE: EXPERIMENTS AND RESULTS

In this thesis, there are two independent machine learning methods employed and explored to solve the problem of protein fold recognition as mentioned in chapter four. One is namely ensemble classifier using hybrid multi-layer back-propagation neural networks (BP- NN), the other is AdaBoost algorithm hybrids to k nearest neighbor classifier. Please note, these two sets of methods are independent and experiments are carried out individually.

Working Datasets

The training and testing datasets in this thesis are taken from Ding and Dubchak (2001) and many research groups have performed their methods on it as discussed in chapter three. The working datasets are available online (http://crd.lbl.gov/~cding/protein/).

The training database, which contains 313 proteins, is based on the PDB_select sets (Hobohm and Sander, 1994), where two proteins have no more than 35% of the sequence identity for the aligned subsequences longer than 80 residues. The independent testing dataset, which contains 385 proteins, is composed of protein sequences of less than 40% identity with each other and less than 35% identity with the proteins of the training dataset. In fact, 90% of the proteins of the testing dataset have less than 25% sequence identity with the proteins of the training dataset (Dubchak *et al*, 1995, 1999). The proteins in the training and testing datasets are classified into the 27 fold types according to the SCOP database (Andreeva *et al, 2004;* Murzin *et al*, 1995).

Table 4. 27 SCOP folds used in current study (source from Dubchak *et al*, 1999)

Index	Structure Class	Fold	Ntrain	Ntest
1	all α	Globin-like	13	6
2		Cytochrome c	7	9
3		DNA-binding 3-helical bundle	12	20
4		4-helical up-and-down bundle	7	8
5		4-helical cytokines	9	9
6		EF-hand	7	9
7	all β	Immunoglobulin-like beta-sandwich	30	44
8		Cupredoxins	9	12
9		Viral coat and capsid proteins	16	13
10		ConA-like lectins/glucanases	7	6
11		SH3-like barrel	8	8
12		OB-fold	13	19
13		Trefoil	8	4
14		Trypsin-like serine proteases	9	4
15		Lipocalins	9	7
16	α/β	(TIM)-barrel	29	48
17		FAD (also NAD)-binding motif	11	12
18		Flavodoxin-like	11	13
19		NAD(P)-binding Rossmann-fold domains	13	27
20		P-loop containing nucleotide triphosphate hydrolases	10	12
21		Thioredoxin-like	9	8
22		Ribonuclease H-like motif	10	14
23		Hydrolases	11	7
24		Periplasmic binding protein-like	11	4
25	$\alpha+\beta$	β -Grasp	7	8
26		Ferredoxin-like	13	27
27		Small inhibitors, toxins, lectins	14	27
Total			313	385

Among the 27 fold types, there are 6 types belong to all α structural class, 9 types to all β class, 9 types to α/β class, and 3 types to $\alpha+\beta$ class. Therefore, the classification of 27 folds is one level deeper and more challenging than that of 4 major structural classes. These 27 fold types and the corresponding number of proteins in training (Ntrain) and testing (Ntest) are shown in Table 4.

The feature vector of the working datasets can be concatenated into six feature subsets, named *amino acids composition, predicted secondary structure, hydrophobicity, normalized van der Waals volume, polarity,* and *polarizability,* respectively. They are all extracted from protein sequences based on the methods of feature extraction which has been described in chapter three. There are 21 feature values in each subset, except *amino acids composition,* in which there are 20 feature values.

Table 5. Six feature subsets extracted from protein sequence and the dimension of the feature vectors (source from Dubchak *et al*, 2001)

Parameter set	**Symbol**	**Dimension**
Amino acid composition	C	20
Predicted secondary structure	S	21
Hydrophobicity	H	21
Normalized van der Waals volume	V	21
Polarity	P	21
Polarizability	Z	21
Total		125

Experiment 1: Ensemble Classifier Hybrid Two-layer BP- NN

In experiment 1, ensemble neural network computation is used for the protein folding classification tasks pursued in this thesis. The organization of this section is as following: (i) system architecture and I/O vectors, illustrating the architecture of this experiment, as well as the input and output vectors of two layers back-propagation neural networks; (ii) class label binary representation, explaining why and how to convert the class labels to binary string; (iii) data preparation, describing how to partition the working datasets for two layers networks training and cross validation; and (iv) classification within six folds, introducing an effort made on six classes which are taken from the original datasets and show the performance as well as the discussion.

System Architecture and I/O Vectors

The hybrid classifier architecture, which has been discussed in chapter four, consists of two layers of back-propagation neural networks (BP-NN). The hybrid intelligent classifiers consist of a set of ensemble networks using back-propagation and the network outputs are summarized by another back-propagation neural network.

At first layer, six BP-NNs are used. The inputs of them are the six feature subsets, *amino acids composition, predicted secondary structure, hydrophobicity, normalized van der Waals volume, polarity*, and *polarizability*. These six neural networks are trained individually and independently. The number of hidden layers and how many neurons in each hidden layer could be different in each of these six networks.

Just as already have been discussed in previous chapter, these parameters can be adjusted depending on the different network configuration and data.

At the second layer of system architecture, a single Back-propagation neural network (BP-NN) implements the fusing stage. It uses the outputs from the first layer neural networks as the input vectors to train the second layer neural network (see figure 3).

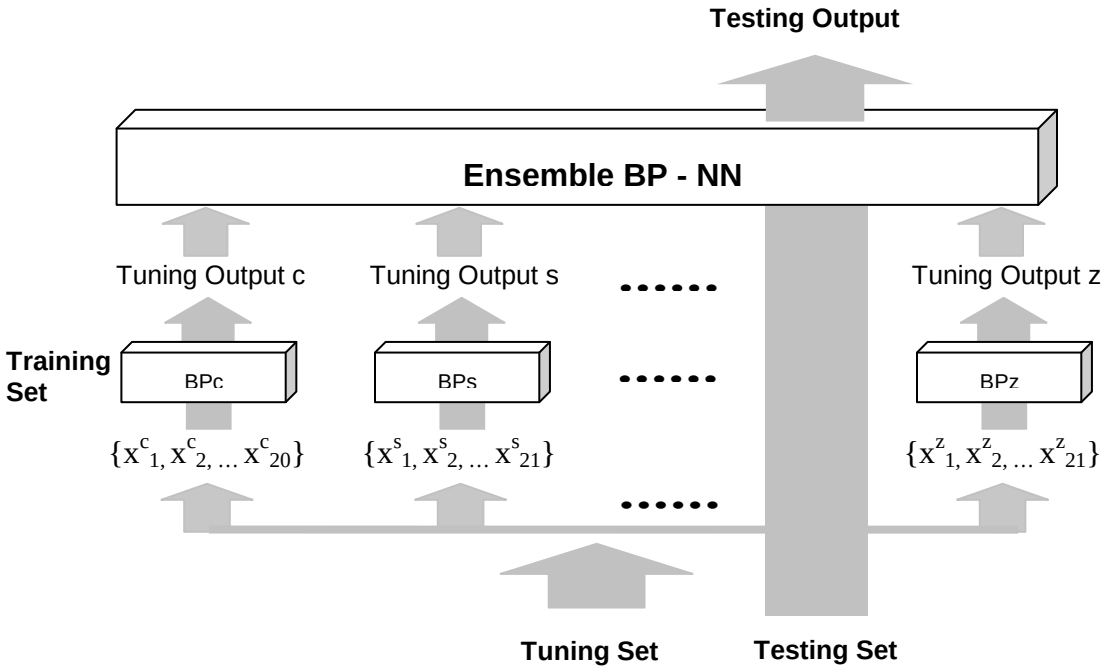

Figure 3. System architecture and I/O vectors of the ensemble classifier hybrid two-layer BP- NN

Class Label Binary Representation

As described before, a single back-propagation neural network, which implements the fusing stage at the second layer of system architecture, uses the first layer's outputs as input vectors to train the second layer's back-propagation neural network. Accordingly, the class labels, which act as target outputs in first layer's networks as well as the inputs in second layer's neural network, should be converted to binary string. Here shows an example of the switching when dealing with a dataset having 5 classes. The original class labels are transferred to binary strings as follows.

Class 1	→	1	0	0	0	0
Class 2	→	0	1	0	0	0
Class 3	→	0	0	1	0	0
Class 4	→	0	0	0	1	0
Class 5	→	0	0	0	0	1

Data Preparation

As shown in figure 3, there are two datasets used to train the back-propagation neural networks at two layers individually. Therefore, the original training set is partitioned into two parts, called training set and tuning set, used to train the first and the second layer neural networks, respectively. With this purpose, the original training set in this experiment, is randomly partitioned into 3 parts, Set1, Set2, and Set3. The partition and usage of these sets are listed in figure 4. (i) Set1 and Set2 act as training set to train the first layer neural networks, (ii) all of these 3 sets are brought into the tuning set to test

the first layer neural networks, and the outputs are used to train the second layer network, and (iii) the unseen testing set are taken to both of these two layers neural networks. To evaluate the robustness of this learning method, 3 fold cross validation is carried out, by repeating the above steps 3 times, using Set1 and Set2 as training set at first time, Set1 and Set3 at second time, and then Set2 and Set3. All of these3 sets are used as tuning set at each time.

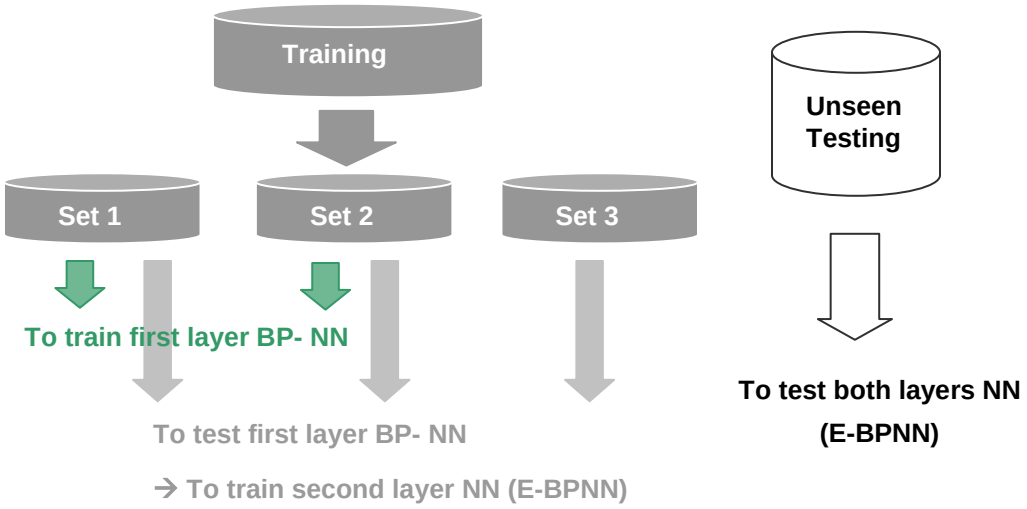

Figure 4. The partition of the original datasets and their usage.

Classification within Six Protein Folds

Before applying this method to the whole working datasets which contains 27 classes, 6 classes are taken from the original working datasets as the advance troops. The

fold name and the number of proteins in original training set and unseen testing set for each of these six folds as well as the number of proteins in Set1, Set2, and Set3 (which are from original training set) are listed in table 6.

Table 6. Six protein folds are taken from the datasets and the original training set are further partitioned into 3 sets.

Fold Name	**# in Original Training**	**# in Set1**	**# in Set2**	**# in Set3**	**# in Test**
Globin-like	13	5	4	4	6
Immunoglobulin-like	30	10	10	10	44
Viral coat and capsid proteins	16	6	5	5	13
(TIM)-barrel	29	9	10	10	48
NAD(P)-binding Rossmann-fold	13	4	5	4	27
Small inhibitors, toxins, lectins	14	4	5	5	27
Total	115	38	39	38	165

The performance of the classification within these six protein folds using this ensemble classifier hybrid two-layer BP- NN is shown in table 7. The successful rate on classifying unseen testing sample are 70%, 67%, and 61% for 3 fold cross validation, respectively. And the successful rate on each of the six classes varies from 46% to 73%

(average within 3 fold cross validation). The differences of the accuracy on each of 3 folds are from 8% (Class 6) to 39% (Class 3). It is rational to believe that there might be much lower accuracy when applying this method to all of the 27 classes in original working datasets, as mentioned in previous chapters. With such concern, we tend to look for and further explore some ensemble approaches which can help to improve the accuracy of the given machine learning algorithms.

Table 7. The performance of the classification within these six protein folds using the ensemble classifier hybrid two-layer BP- NN. A 3-fold cross validation is carried out.

	Number of Test Samples	Correct Identification			Average	Standard Deviation
		CV1	CV2	CV3		
Class 1	6	4 (67%)	4 (67%)	3 (50%)	3.7 (61%)	0.58
Class 2	44	32 (73%)	28 (64%)	29 (66%)	29.7 (67%)	2.08
Class 3	13	7 (54%)	8 (62%)	3 (23%)	6.0 (46%)	2.65
Class 4	48	33 (69%)	38 (79%)	34 (71%)	35.0 (73%)	2.65
Class 5	27	19 (70%)	13 (48%)	12 (44%)	14.7(54%)	3.79
Class 6	27	21 (78%)	19 (70%)	19 (70%)	19.7 (73%)	1.15
Total	165	116 (70 %)	110 (67%)	100 (61%)	109 (66%)	8.08

Experiment 2: AdaBoost Algorithm Hybrids K Nearest Neighbor Classifier

AdaBoost is such kind of method, with the goal of improving the accuracy of any given learning algorithm by incrementally adding one hypothesis at a time to an ensemble. Each new hypothesis is constructed by a learning algorithm that seeks to minimize the classification error on a weighted training data set. In experiment 2, it is employed to hybrid the baseline algorithm, k nearest neighbor algorithm. The reason of choosing k nearest neighbor as our baseline algorithm is that among the various methods of supervised statistical pattern recognition, normally nearest neighbor algorithm achieves comparatively stable performance with the only parameter k to be set. Therefore, it would be easy to tell whether AdaBoost can make contribution on this classification task. Under such intention, we would like to make the system architecture straightforward in this experiment by putting six feature subsets, *amino acids composition*, *predicted secondary structure*, *hydrophobicity*, *normalized van der Waals volume*, *polarity*, and *polarizability* together to form 125 dimensional feature vectors ($20 + 21 \times 5$) and learning them as a whole piece. Moreover, there is no need to further normalize these features due to the reason that all of them are percent composition, percent frequencies, or distribution, with the values from 0 to 1.

To demonstrate the power of our classifier, predictions are conducted based on the same training and testing datasets which were used by the previous investigators as described in chapter three. None of proteins in these datasets has more than 35% sequence identity to any other, and most of proteins in the testing dataset have below 25% sequence identity with those in the training dataset (Ding & Dubchak, 2001).

The organization of this section is as following: (i) cross validation within the training dataset, illustrating how to carry out 5 fold cross validation within the training dataset to evaluate the robustness of this approach and choose the best classifier for unseen testing samples; (ii) prediction on unseen testing set, showing the performance of classifying test proteins into 27 classes and with the comparison to the benchmark.

Cross Validation within Training Dataset

The purpose of cross validation in this experiment is to choose the best classifier for the unseen testing set by determining the values of the parameters in the learning algorithms used in this experiment, and to evaluate the strength and consistency of this method.

According to the two algorithms, AdaBoost and k nearest neighbor, employed in this experiment, there are two parameters should be determined before using the classifier to do classifying on the independent testing dataset. One is *k* (the number of neighbors) in weak classifier using k nearest neighbor algorithm, the other is the iteration number in AdaBoost algorithm (here, we name it *i*). To successfully choose these two parameters, we use cross validation within the original training dataset and compare the performances of the classifiers using different *k* and *i*. A 5-fold cross validation is carried out within the original training set, which contains 313 proteins classified into 27 types of protein folds. The original training set is partitioned into 5 subsets; each one contains 62 or 63 proteins. Of the 5 subsets, a single subset is retained as the validation data for testing the model, and the remaining 4 subsets are used as training data. The cross validation

process is then repeated 5 times, with each of the 5 subsets used exactly once as the validation data.

Table 8. Accuracy (in percents) for AdaBoost hybrid k nearest neighbor algorithm, depending on different values of *k* and *i*. A 5-fold cross validation is carried out to get the average accuracy.

	k = 1	k = 2	k = 3	k = 4	k = 5
i = 0	0.631	0.625	0.610	0.619	0.615
i = 5	0.560	0.551	0.495	0.500	0.560
i = 10	0.455	0.522	0.421	0.415	0.418
i = 15	0.419	0.480	0.406	0.413	0.414
i = 20	0.438	0.439	0.397	0.410	0.405
i = 25	0.435	0.425	0.382	0.397	0.402
i = 30	0.430	0.405	0.382	0.375	0.398
i = 35	0.425	0.410	0.375	0.360	0.395
i = 40	0.418	0.415	0.362	0.365	0.375
i = 45	0.410	0.398	0.349	0.361	0.371
i = 50	0.396	0.380	0.349	0.356	0.368
i = 55	0.410	0.385	0.353	0.368	0.362
i = 60	0.422	0.380	0.349	0.363	0.354
i = 65	0.401	0.391	0.353	0.356	0.348
i = 70	0.398	0.402	0.349	0.347	0.380
i = 75	0.401	0.388	0.349	0.355	0.383
i = 80	0.405	0.372	0.349	0.368	0.395

The average error rates are given in Table 8 depending on various values of *k* and *i* when 125 dimensional feature vectors are used and a 5-fold cross validation is carried out. The value of *k* to be tested and compared is set from 1 to 5 here since the larger *k*, the more outliers have a chance to be included into a neighborhood. And we stop the iteration when it arrives 80. From table 8, one can see that the error rate is already below 0.4 after 20 iterations when $k = 3$. Such performance is encouraging because that there is low sequence identity between any two proteins in the training set.

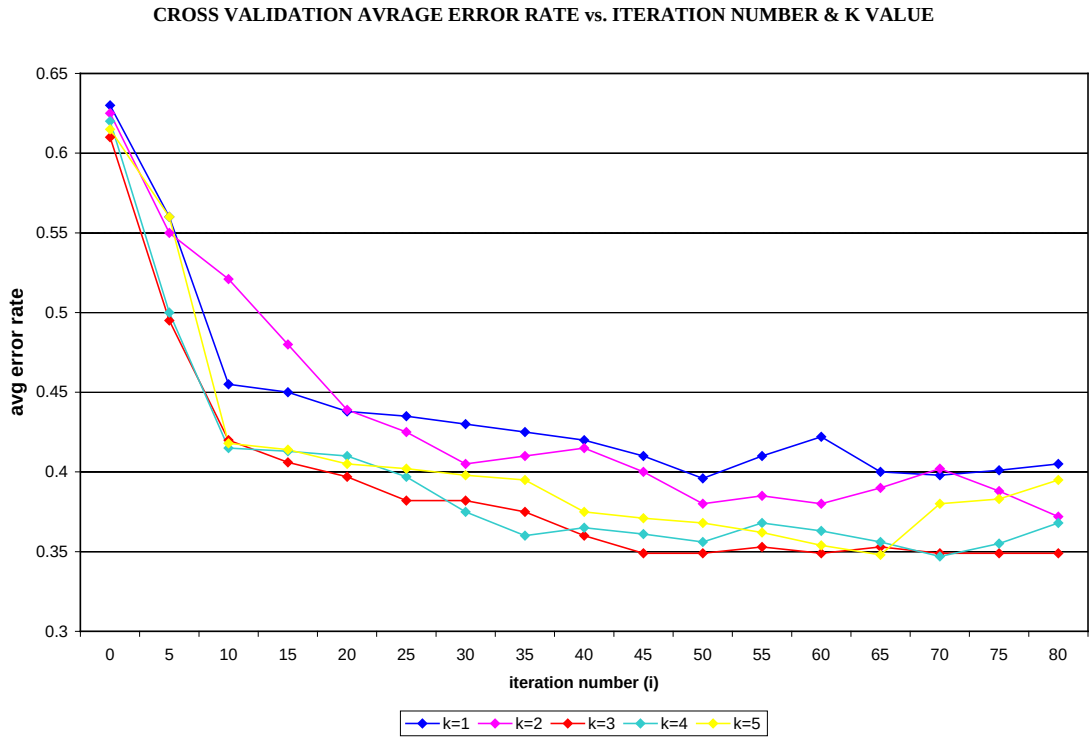

Figure 5. Line chart of the cross validation average accuracy rates depending on different *k* values and iteration numbers

Figure 5 is the line chart of the average accuracy rates. From the line chart one can see that the overall error rate drops down with respect to the increment of *i* values especially from *i* = 0 to 50 and there is small fluctuation from 50. Based on it, we can say that the AdaBoost algorithm does help to improve the accuracy of k nearest neighbor, the baseline algorithm used in this experiment, as we have expected. Moreover, the line for *k* = 3 (which is in red in figure 5) drops down steadily with the comparison to other lines for different *k* values. Therefore, *i* = 50 and *k* = 3 might be the good choice for the classifier which will be used on the unseen validation set.

Prediction on Unseen Testing Set

Predictions are conducted on the independent unseen testing datasets which contains 27 types of folds and where most of proteins have below 25% sequence identity with those in the training dataset.

To illustrate the contribution of AdaBoost on this task, figure 6 shows both the error rate of prediction on test samples by employing k nearest neighbor with AdaBoost (which is in red) and the error rate by k nearest neighbor classifier individually (which is in blue) when *k* = 3. One can see that the error rate of k-NN classifier with AdaBoost is always below the one without using AdaBoost algorithm, which can validate the assumption and support the motivation of the methods used in this experiment.

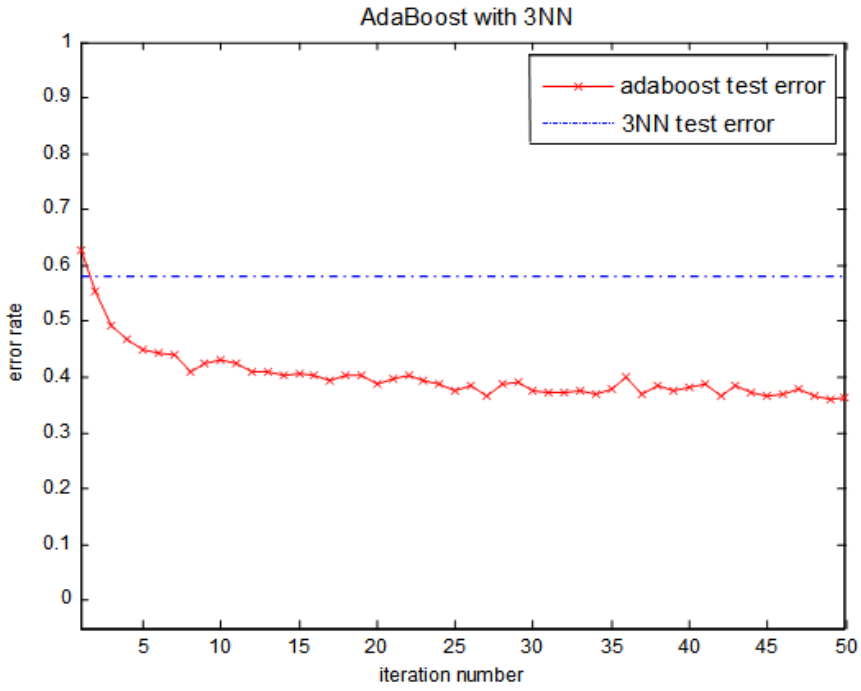

Figure 6. Error rate of prediction on test samples using k- NN classifier with AdaBoost (in red) vs. k- NN classifier without AdaBoost (in blue) when k = 3

The overall success rate in recognizing the fold among the 27 folding types by our classifier, with i = 50 and k = 3 (determined through cross validation), for the proteins in the independent dataset is given in Table 9, where, for facilitating comparison, the success rates by the other approaches are also listed. From Table 9, one can see that our classifier, which is formed by AdaBoost algorithm hybrids to k nearest neighbor classifiers (KNN-AB), successfully compete even outperform the other approaches.

Table 9. Overall accuracy by different approaches in recognizing the fold types for proteins in the independent testing dataset

Classifier	Source	Accuracy
NN (Neural Networks) [a]	Ding and Dubchak, 2001	41.8%
SVM (Support Vector Machine) [b]	Ding and Dubchak, 2001	45.2%
SVM (Support Vector Machine) [c]	Ding and Dubchak, 2001	51.1%
SVM (Support Vector Machine) [d]	Ding and Dubchak, 2001	56.0%
BIMLP-B (Discretized Interpretable Multi-Layer Perceptrons with Bagging)	Bologna and Appel, 2002	61.2%
BIMLP-A (Discretized Interpretable Multi-Layer Perceptrons with Arcing)	Bologna and Appel, 2002	59.1%
MLP (Multi-Layer Perceptron)	Chung and Huang, 2003	48.8%
RBFN (Radial Basis Function Network)	Chung and Huang, 2003	49.4%
GRNN (General Regression Neural Network)	Chung and Huang, 2003	44.2%
SVM (Support Vector Machine)	Chung and Huang, 2003	51.4%
HKNN (K-Local Hyperplane Distance Nearest Neighbor algorithm)	Okun, 2004	57.4%
KNN-AB (K-Nearest Neighbor algorithm with AdaBoost)	**Yijing, 2007**	**64.7%**

[a] The training method for NN is “one against others”.

[b] The training method for SVM is “one against others”.

[c] The training method for SVM is “unique one against others”.

[d] The training method for SVM is “all against all”.

Table 10 lists the prediction accuracy (in percentage) for each individual class of 27 folds with the benchmark performance comparison. Figure 7 is the bar chart of the prediction accuracy for individual fold by three different methods used on the same working datasets. The blue bars are the prediction success rate using KNN-AB (AdaBoost algorithm hybrids to k nearest neighbor classifiers) in this experiment; the yellow bars are the success rate using SVM (Support Vector Machine) reported in Ding & Dubchak’s paper (2001); and the red bars are the success rate using HKNN (K-Local Hyperplane Distance Nearest Neighbor algorithm) reported in Okun’s paper (2004). One can see that among the 27 prediction accuracy on individual class, KNN-AB has the highest accuracy on 18 classes and only 3 of 27 are lower.

Table 10. Prediction accuracy (in percentage) for each individual fold and overall accuracy (bottom line) for 27 folds

Fold Index	SVM [d] (Ding & Dubchak, 2001)	HKNN (Okun, 2004)	**KNN-AB (Yijing, 2007)**
1	83.3	83.3	100.0
2	77.8	77.8	88.9
3	35.0	50.0	60.0
4	50.0	87.5	100.0
5	100.0	88.9	77.8
6	66.7	44.4	22.2
7	71.6	56.8	72.7
8	16.7	25.0	50.0
9	50.0	84.6	76.9
10	33.3	50.0	50.0
11	50.0	50.0	50.0
12	26.3	42.1	36.8
13	50.0	50.0	75.0
14	25.0	50.0	25.0
15	57.1	42.9	42.9
16	77.1	79.2	95.8
17	58.3	58.3	75.0
18	48.7	53.9	38.5
19	61.1	40.7	51.9
20	36.1	33.3	41.7
21	50.0	37.5	50.0
22	35.7	71.4	83.3
23	71.4	71.4	85.7
24	25.0	25.0	75.0
25	12.5	25.0	50.0
26	37.0	25.9	44.4
27	83.3	85.2	64.0
Average	56.0	57.1	64.7

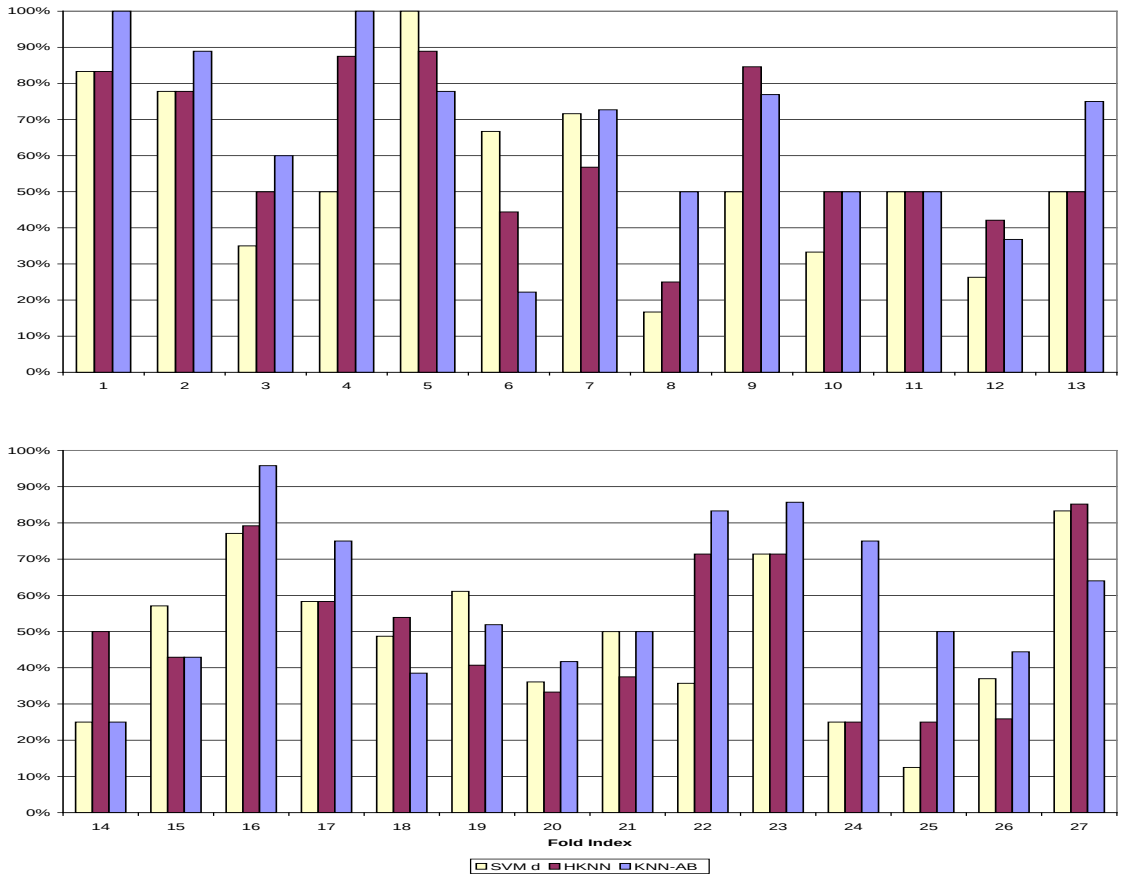

Figure 7. Bar chart of the prediction accuracy for individual fold by three different methods used on the same working datasets.

CHAPTER SIX: DISCUSSION

In this thesis, two computational methods are investigated and explored for the task of protein fold recognition. One is the ensemble classifier hybrid multi-layer back-propagation neural networks; the other is AdaBoost algorithm hybrid to k nearest neighbor classifier. These two methods are applied on a real-world dataset individually.

To carry out these computational methods, we take advantage of the information extracted from structural and physico-chemical properties beyond sequence information, which forms 6 feature subsets and 125 features in total. Two independent experiments are set up to perform these two methods individually. The results show that the approach of AdaBoost algorithm hybrids to k nearest neighbor classifier achieves comparatively better performance, which reaches 64.7% successful rate in classifying independent validation dataset into 27 types of protein fold. The following three sections briefly summarize the merit and contribution of this method, discuss the limitation of this method, as well as the future work.

Summary

AdaBoost algorithm is an effective method for constructing a “strong” classifier as linear combination of “weak” classifiers. Since introduced by Freund and Schapire in 1995, AdaBoost has successfully solved many practical problems in different fields, including some problems in bioinformatic research, such as protein subcellular localization and promoter detection. However, it has never been applied onto the task of protein folding pattern recognition. In this thesis, the experiment proves the merit of this

approach, as it shows the performance of this method is superior to all the existing published researches based on fair competition – using identical sample sets, feature representation, and class labels. The benchmark study shows that AdaBoost strategy coupling with weak learning classifier lead to improved and robust performance of 64.7% recognition rate versus 41.8% - 61.2% accuracy reported in published literatures. Besides, the result of experiment shows that the performance of k nearest neighbor classifier with AdaBoost algorithm is always better than of k nearest neighbor baseline algorithm without AdaBoost for the all iterations. It validates the contribution of AdaBoost algorithm and supports our motivation of applying AdaBoost to improve the accuracy of k nearest neighbor algorithm. Furthermore, this method has other advantages, such as the shorter time that spent on the whole process of training and testing and the fewer adjustable parameters compared to Support Vector Machines or Neural Networks.

The experimental procedure and setup in this thesis was well-designed. The experiment framework consists of two steps, cross validation within the training dataset and prediction on unseen validation dataset. A 5-fold cross validation is carried out with the purpose of choosing the best value of the parameters with good grounds and evaluating the consistency of merit of AdaBoost algorithm hybrids to k nearest neighbor classifier. Accordingly, we keep track of all the weaker classifiers during iterative adaptation and apply the best classifier on the unseen testing samples. Moreover, this thesis systematically evaluates the robustness of the approach compared with the performance reported by other research groups which have performed their efforts on the same working datasets.

Limitations

As discussed in previous chapter, AdaBoost gives each training sample a different weight in each iteration by updating the weights dynamically according to the errors in previous learning. In this way, AdaBoost focuses in on the difficult patterns to greedily minimize the errors. Simultaneously, it makes AdaBoost has a tendency to overfit when there is significant noise in the training data, preventing it from learning an effective ensemble (Dietterich, 2000). In the experiment, we find that AdaBoost strategy is able to correctly classify a large percentage of data in most iteration, but struggle for better estimating on a small group of samples. Furthermore, the weight values for the hard samples grow as the iteration number increases, making those proteins get more chances to be selected in the re-sampling process. Accordingly, AdaBoost is "trapped" in those data.

With such concern, therefore, we set the samples with the highest weights to be 0 after *n* iterations. By doing this, we could discard those hard examples (noise) and give AdaBoost chances of jumping out of these noisy data. Besides, it is very important to carefully preprocess the data with the purpose of removing the noise. It might achieve even better performance when using some of 6 feature subsets and 125 features, instead of using all of them. However, de-noise is always tradeoff with retaining meaningful biological interpretation.

Future Work

From the results of experiment 2, one can obviously see the contribution of

AdaBoost algorithm to the improved accuracy of the baseline algorithm, k nearest neighbor. As mentioned before, the reason of selecting k nearest neighbor in this experiment is its comparatively stable performance among the various methods of supervised statistical pattern recognition, which would make it easy to tell whether AdaBoost can make contribution on this protein fold recognition task. Since the result of the experiment in this thesis strongly prove the merits of AdaBoost algorithm, we would like to apply AdaBoost to hybrid with more sophisticated learning algorithms, such as Support Vector Machines or Neural Networks. Both of these two algorithms have been widely used in bioinformatic research including protein folding pattern recognition. Therefore, the upcoming goal is to find a feasible and efficient way to combine AdaBoost and Support Vector Machine or Neural Networks with intention of dealing with multiple class label classification when the number of classes is large. It would be valuable to the task of protein fold recognition and other bioinformatic research.

REFERENCES

Andreeva, A., Howorth, D., Brenner, S. E., Hubbard, T. J., Chothia, C. and Murzin, A. G. (2004) SCOP database in 2004: refinements integrate structure and sequence family data. *Nucleic Acids Res*, vol. 32, pp. 226-229.

Baldi, P. and Brunak, S. (2001), *Bioinformatics: the Machine Learning Approach*, MIT Press.

Bauer, E. and Kohavi, R. (1999), *An empirical comparison of voting classification algorithms: bagging, boosting, and variants, Machine Learning*, vol. 3, pp.105-142.

Bologna, G. and Appel, R.D. (2002), A comparison study on protein fold recognition. *Proc. of the 9th Int. Conf. on Neural Information Processing, Singapore*, pp. 2492-2496.

Branden, C. and Tooze, J. (1991) *Introduction to protein structure*, Garland, New York

Chung, I.-F., Huang, C.-D., Shen, Y.-H., and Lin, C.-T. (2003), Recognition of structure classification of protein folding by NN and SVM hierarchical learning architecture. *Lecture Notes in Computer Science: Artificial Neural Networks and Neural Information Processing* (*ICANN/ICONI*), by Kaynak, O., Alpaydin, E., Oja, E., and Xu, L. (Eds.), Springer-Verlag, Berlin, vol. 2714, pp. 1159-1167.

Denoeux, T. (1995), A k-nearest neighbor classification rule based on Dempster- Shafer theory. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, pp. 804-813.

Dietterich, T.G. (2000), *An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization, Machine Learning*, vol. 40, pp. 139-157.

Ding, C.H.Q. and Dubchak, I. (2001), Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics*, vol.17 (4), pp. 349-358.

Dubchak, I., Muchnik, I., Holbrook, S. R. and Kim, S. H. (1995) Prediction of protein folding class using global description of amino acid sequence. *Proc Natl Acad Sci U S A*, vol. 92, pp.8700-8704.

Dubchak, I., Muchnik, I., Mayor, C., Dralyuk, I. & Kim, S. H. (1999) Recognition of a protein fold in the context of the Structural Classification of Proteins (SCOP) classification. *Proteins*, vol. 35, pp. 401-407.

Freund, Y. and Schapire, R.E. (1996). Experiments with a new boosting algorithm. In *13th International Conference on Machine Learning*, San Francisco. pp.148-146.

Freund, Y. and Schapire, R.E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and Systems Sciences*, vol. 55, pp. 119-139.

Haykin, S. (1998) *Neural networks: a comprehensive foundation*, Prentice Hall

Hobohm, U. and Sander, C. (1994), Enlarged representative set of proteins, *Protein Sci*., vol. 3, pp. 522-524.

Kuncheva, L. I., Bezdek, J. C., Duin, R. P. W. (2001), Decision templates for multiple classifier fusion. *Pattern recognition*, vol. 34(2), pp. 299-314.

Lesk, A.M. (1991) *Protein architecture: a practical approach*, IRL press, Oxford

Malacinski, G. (2003) *Essentials of molecular biology*, Jones and Bartlett Publishers, Inc.

Murzin, A. G., Brenner, S. E., Hubbard, T. and Chothia, C. (1995) SCOP: a structural classification of protein database for the investigation of sequence and structures. *Journal of Molecular Biology*, vol. 247, pp. 536-540.

Okun, O. (2004), Protein fold recognition with k-local hyperplane distance nearest neighbor algorithm. In *Proc. of the 2nd European Workshop on Data Mining and Text Mining for Bioinformatics, Pisa, Italy*.

Oza, N. C., Polikar, R., Kittler, J., Roli, F. (2005), *Multiple classifier systems*, Springer, Berlin

Pevzner, P (2000) *Computational molecular biology: an algorithmic approach*, MIT Press

Rost, B. and Sander, C. (1994), Combining evolutionary information and neural networks to predict protein secondary structure. *Proteins*, vol. 20, pp. 216-226.

Shi, S.Y.M. and Suganthan, P.N. (2003), Feature analysis and classification of protein secondary structure data *Lecture Notes in Computer Science: Artificial Neural Networks and Neural Information Processing* (*ICANN/ICONI*), by Kaynak, O., Alpaydin, E., Oja, E., and Xu, L. (Eds.), Springer-Verlag, Berlin, vol. 2714, pp.1151-1158.

Tomii, K. and Kanehisa, M. (1996), Analysis of amino acid indices and mutation matrices for sequence comparison and structure prediction of proteins. *Protein Eng.*, vol. 9, pp. 27-36.

VITA

Yijing Su

yijing.su@caremark.com
(920) 698-6105
400 Symphony Circle
Cockeysville, MD 21030

Education

Master of Science in Bioinformatics, Expected December 2007
School of Informatics, Indiana University Purdue University at Indianapolis
Thesis: Protein Fold Recognition Using AdaBoost Learning Strategy
Advisor: Jeffrey Huang

Bachelor of Medicine in Clinical Medicine, June 2000
Medical Center, Fudan University

Research Interests

- Protein fold prediction
- Machine learning

Experiences

Data Analyst, Product Development & Training, CVS Caremark, Hunt valley, MD
June 2007 - Present
- Established different scenarios and tested analytic methods using SAS and Oracle.

System Analyst, Data Warehousing & Analytic, Kohler Company, Kohler, WI
May 2006 – September 2006
- Manipulated data to generate reports using SAS and BI tools.
- Built data model using Erwin Data Modeler.

Faculty Assistant, School of Informatics, IUPUI, Indianapolis, IN
August 2005 - December 2005
- Generated reports on research of social foundation and impact of informatics.

Resident, Department of Medical Imaging, Zhongshan Hospital, Shanghai, China
June 2000 – June 2002