國 立 交 通 大 學

電機與控制工程學系

博 士 論 文

資料融合及山峰群聚法應用於改善蛋白質結構預測與分析

Improvement of Protein Structure Prediction and Analysis by Data Fusion and Mountain Clustering Approaches

研 究 生： 林 肯 豊
指導教授： 林 進 燈 教授

中 華 民 國 九 十 七 年 九 月

資料融合及山峰群聚法應用於

改善蛋白質結構預測與分析

Improvement of Protein Structure Prediction and Analysis by Data Fusion and Mountain Clustering Approaches

研　究　生：林肯豊	Student：Ken-Li Lin
指導教授：林進燈　博士	Advisor：Dr. Chin-Teng Lin

國立交通大學

電機與控制工程學系

博士論文

A Dissertation

Submitted to Department of Electrical and Control Engineering

College of Electrical Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Electrical and Control Engineering

September 2008

Hsinchu, Taiwan, Republic of China

中　華　民　國　九　十　七　年　九　月

資料融合及山峰群聚法應用於
改善蛋白質結構預測與分析

研究生：林肯豐　　　　　　　　　　　指導教授：林進燈　博士

國立交通大學　電機與控制工程學系　博士班

摘　要

在本研究中，主要探討二個與蛋白質結構預測與分析相關的問題，首先，將類神經網路以二階段分類的階層式學習架構用於蛋白質的結構預測分類問題，並進一步延伸，結合資訊融合的組合分析技術，有系統的利用多樣性次序/計分圖 (Diversity rank/score graph)，選取出重要的分類特徵，藉以提昇第一階段及第二階段的分類正確率分別達到87%及69.6%，印證此組合特徵擷取方式及系統分類架構，確為有效的方法，可協助改善此類蛋白質結構預測分類的問題，提昇正確率。其次，利用山峰群聚法來分析蛋白質3D結構的組成區塊，結合Best Molecular Fit (BMF)方法用於計算3D結構距離，使傳統山峰群聚法，可轉而用於立體三維空間向量之分群(稱之為Structural Mountain Clustering Method, 簡稱SMCM)，藉由估測區域密度來找出有用的3D結構組成區塊，並以實例驗證當這些組成區塊用於重建蛋白質3D結構時，以整體及區段均方根誤差值(Global-fit Root Mean Square Error及Local-fit Root Mean Square Error) 作為衡量標準時，均獲得良好的效果。另外，也針對SMCM山峰群聚法進行計算複雜度的探討，並提出遞增法(Incremental approach)來運用山峰群聚法，以因應一次處理大量訓練資料時，計算複雜度高而耗時過久的情形，此外，文中也採用不同的效能評比方式，以實例驗證本方法較以往二階段群聚法有更好的效果。

Improvement of Protein Structure Prediction and Analysis by Data Fusion and Mountain Clustering Approaches

Student: Ken-Li Lin Advisor: Dr. Chin-Teng Lin

Department of Electrical and Control Engineering

National Chiao Tung University

Abstract

In this dissertation, we focus on two issues concerning protein structure prediction and analysis. First, we have applied a two-level classification strategy called hierarchical learning architecture (HLA) using neural networks to differentiate proteins according to their classes and folding patterns and then use a combinatorial fusion technique to facilitate feature selection and combination for improving predictive accuracy in protein structure classification. When applying combinatorial fusion to the protein fold prediction problem using neural networks with HLA, the resulting classification has an overall prediction accuracy rate of 87% for four classes and 69.6% for 27 folding categories. These rates are higher than previous results and it demonstrates that data fusion is a viable method for feature selection and combination in the prediction and classification of protein structure.

Second, we propose an algorithm named Structural Mountain Clustering Method (SMCM) to find a library of short 3-D structural motifs (building blocks) for construction of 3-D structures of proteins/peptides. The algorithm finds the building blocks based on an estimate of local "density" of 3-D fragments computed using a measure of structural similarity that is obtained after best molecular fit alignment of pairs of fragments. The algorithm is tested on two well known benchmark datasets and is found to successfully reconstruct the test peptides in terms of both global-fit Root-Mean-Square (RMS) errors and local-fit RMS errors. The good local-fit RMS errors achieved by SMCM indicate that these short structural motifs extracted by our algorithm can model the nearby fragments quite accurately. We then analyze the computational complexity of the SMCM and propose an incremental version of SMCM to deal with large training dataset. In addition to using the global-fit and local-fit RMS errors, we propose and use two alternative ways to compare the quality of such quantization and reconstruction results between SMCM and Two Stage Clustering Algorithm (TSCA) to show the superiority of SMCM.

誌謝

感謝在博士班過程中, 指導老師林進燈教授之引導及協助, 以及 Dr. Pal 在研究方法的指點, 還要謝謝研究上的合作夥伴們（俊淵、立偉、志賢、翊方、淳德、勝富…）以及家人的支持, 謹此誌謝

目錄

表目錄

圖目錄

1. Introduction

1.1 Statements of the Problems

The structure prediction and classification of proteins plays a very important role in bioinformatics, since three-dimensional (3-D) structure of a protein is essential for understanding and studying its function. A lot of efforts have been put by researchers to find the relations between protein sequences and their three-dimensional structures but it's still a difficult and unsolved problem. There are several famous classification databases such as Structure Classification of Protein (SCOP) [1], Class, Architecture, Topology, and Homologous superfamily (CATH) [2], DIAL-derived domain database (DDBASE) and Pfam [3], which imbue the structure with context and analysis. However, the number of known 3-D protein structures is much less than that of the determined protein sequences. Thus, there is still the need for some effective methods to investigate the protein structure from its primary sequence. Finding the 3-D structure of a protein using X-ray crystallography or by nuclear magnetic resonance imaging is not only time consuming but also quite expensive and hence alternative computational approaches are being tried. Computational structure prediction methods will provide valuable information for the large fraction of sequences whose structures will not be determined experimentally. The first class of protein structure prediction methods, including threading or fold recognition and comparative modeling, rely on detectable similarity spanning most of the modeled sequence and at least one known structure. The second class of methods, de novo or ab initio methods, predict the structure from sequence alone, without relying on similarity at the fold level between the modeled sequence and any of the known structures [4].

Among the former methods, 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. There is, therefore, a good chance that a protein which has a similar fold to the target protein has already been studied by X-ray crystallography or NMR spectroscopy and can be found in the PDB (Protein Data Bank). The basic idea is that the target sequence (the protein sequence for which the structure is being predicted) is threaded through the backbone structures of a collection of template proteins. Fold recognition methods can utilize the profile information derived from properties of amino acid sequences and the structures in the fold library and even take into account the local secondary structure (e.g. whether the amino acid is part of an alpha helix) or even evolutionary information (how conserved the amino acid is) for structure prediction. Previous research [5], [6] have shown that an accuracy rate of 70-80% has been achieved to classify most of proteins into four classes according to their amino acid sequence information (i.e., all-alpha (α), all-beta (β), alpha/beta (α/β) and alpha+beta ($\alpha+\beta$)) [1]. In summary, these four classes contain 82.5% folding patterns, 84.7% superfamilies and 88.1% families in the SCOP database (SCOP release version 1.65 [7]). In [8], Ding and Dubchak proposed a taxonmetric approach for protein folding classification (into 27 folding patterns) beyond four simple classes had the highest overall prediction accuracy rate at 56.5%. In Huang et al. [9], extra features were defined and improved the prediction accuracy rate by 9% to reach 65.5%. In this dissertation, we use a combinatorial fusion technique to facilitate feature selection and combination for improving predictive accuracy in this problem and obtain better prediction accuracy rate of 87% for four classes and 69.6% for 27 folding categories.

For the ab initio structure prediction, to predict three-dimensional protein structures from amino-acid sequences alone is a long-standing challenge in computational molecular biology. Since the search space is enormous even for proteins with moderate sequence lengths, the modeling of a protein structure de novo without using templates is quite difficult. To allow rapid and efficient searching of conformational space, often only a subset of the atoms in the protein chain is represented explicitly. Recently, methods based on assembly of short fragments have shown a great promise [10]. Among these methods, 3-D building blocks approaches have been proposed to use a set of proteins with known 3-D structures, first construct libraries of building blocks or short structural motifs, the structures that appear frequently and have some sequence to structure relation. These building blocks are then used to construct or analyze structures of new proteins. The short structural fragments that recur across different protein families can often be viewed as stand-alone units which fold independently and hence can help assignment of building blocks to unknown proteins for reconstruction of 3-D structures [11]. Extraction of good representative building blocks is the key to the success of such approaches. Unger et al. [12] proposed a two-stage clustering algorithm to choose hexamers (fragments of length 6) with a large number of neighbors to be the centers of clusters and hence building blocks. A similar approach was used by Micheletti et al. who considered the largest number of nearby points within a similarity cutoff called "proximity score" [13] to select cluster centers. Kolodny et al., on the other hand, used a simulated annealing k-means method to extract clusters with the minimal total variance score [14]. In this dissertation, a modified form of the mountain clustering / subtractive clustering method [15]-[16] is proposed to find building blocks. Results of some preliminary investigation are reported in [17]. The use of the modified mountain clustering method is computationally expensive when the training data set size is large. To reduce the computational burden, we also propose an incremental version of the structural mountain clustering method. Our experiments with some benchmark datasets show that the proposed algorithms can find better representative building blocks than the method in [12] that selects cluster centers by counting neighbors. We also propose two alternative ways for displaying the quality of the building blocks.

1.2 Organization of the Dissertation

This dissertation is organized as follows. Chapter 2 describes preliminaries and material including the Structural Classification of Proteins (SCOP), 3D building block approach, features used for classification and the datasets used in the dissertation. Chapter 3 explores the results and methods by data fusion approach for the prediction of protein folds. Chapter 4 investigates how to find the useful building blocks for construction of protein structures using a structural variant of Mountain clustering methods. Chapter 5 develops the incremental version of Structural Mountain clustering methods. At last, we make the conclusions in Chapter 6.

2. Preliminaries and Materials

2.1 Structural Classification of Proteins (SCOP) database

The SCOP database is a comprehensive ordering of all proteins of known structure, according to their evolutionary and structural relationships. Protein domains in SCOP are grouped into species and hierarchically classified into families, superfamilies, folds and classes. The first two levels, family and superfamily, describe near and distant evolutionary relationships; the third, fold, describes geometrical relationships. It is originally published in 1995[1] and constantly updated over years till now [1][7][18]-[20]. The database and its associated files are freely accessible from a number of WWW sites mirrored from URL http://scop.mrc-lmb.cam.ac.uk/scop/.

The classification of the proteins in SCOP is on hierarchical levels as follows:

Family. Proteins are clustered together into families on the basis of one of two criteria that imply their having a common evolutionary origin: first, all proteins that have residue identities of 30% and greater; second, proteins with lower sequence identities but whose functions and structures are very similar; for example, globins with sequence identities of 15%.

Superfamily. Families whose proteins have low sequence identities but whose structures and, in many cases, functional features suggest that a common evolutionary origin is probable, are placed together in superfamilies; for example, the variable and constant domains of immunoglobulins.

Common fold. Superfamilies and families are defined as having a common fold if their proteins have the same major secondary structures in the same arrangement and with the same topological connections. The structural similarities of proteins in the same fold category probably arise from the physics and chemistry of proteins favoring certain packing arrangements and chain topologies.

Class. The different folds have been grouped into classes. Most of the folds are assigned to one of the five structural classes:

1. all-α, those whose structure is essentially formed by α-helices;
2. all-β, those whose structure is essentially formed by β-sheets;
3. α/β, those with α-helices and β-strands;
4. α+β, those in which α-helices and β-strands are largely segregated;
5. multi-domain, those with domains of different fold and for which no homologues are known at present.

Other classes have been assigned for peptides, small proteins, theoretical models, nucleic acids and carbohydrates.

Following the previous published papers [5], [8], [21], we focus on the first four main classes, i.e. all alpha (α), all beta (β), alpha and beta (α+β) and alpha/beta (α/β), with 27 folds according to their structures representing all major structural classes.

2.2 Feature Sets used for Predicting Protein Folds

2.2.1 Global Features

In the previous studies [5], [8], several features have been considered for predicting protein folds by using global descriptors computed from the physical, chemical or structural properties of the constituent amino acids. Each property of the sequence was described based on three global descriptors: Composition (C), Transition (T), and Distribution (D) [5], [8]. The descriptor "Composition" is the occurrence percentage of each characteristic (attribute). The descriptor “Transition” is change count from one characteristic to another. There will be different combinations which would be C_2^m for m kinds of characteristics. The descriptor “Distribution” is the percentage of m kinds of characteristics appearing at the location in 0%、25%、50%、75%、100% of the sequence. These descriptors for each characteristic essentially describe the frequencies with which the properties change along the sequence and their distribution on the chain including the rate of composition, transition and distribution. These properties of the amino acids after encoded by the descriptors were used as input features to the machine learning network.

Next, we would like to illustrate how these descriptors are computed. For example, the hydrophobicity of an amino acid could be classified into 3 types: “positive”, “neutral” and ”negative”. If we denote the three types of characteristics by “A”, “B”, “C” alphabets, then each of amino acids in the protein sequence can be replaced by these three alphabets, and therefore a new sequence represented by A、B、C is obtained. Table 1 shows such an example sequence that each position of individual alphabet are numbered. Then we can count the percentages of A、B and C for the whole sequence and obtain the composition as shown in the first row of Table 2. Also we can calculate the transition count for the pairs of A / B, B/ C and A / C and compute the values in the second row o f Table 2. Finally, we calculate the position of 0%、25%、50%、75%、100% of A, B and C, and we can get 15 (5x3=15) distribution values from the ABC sequence as shown in the bottom rows of Table 2. After the procedure, a feature vector of dimension 21 (3+3+5+5+5) can be obtained for this specific property encoded.

Table 1. An example sequence of characteristics using 3 alphabets.

Sequence	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	A	A	B	B	C	A	B	A	B	C	A	A	A	B	B	B	C	A	B	C
	A A	A B	B B	B C	C A	A B	B A	A B	B C	C A	A A	A A	A B	B B	B B	B C	C A	A B	B C	
A	1	2				3		4			5	6	7					8		
B			1	2			3		4					5	6	7			8	
C					1					2							3			4

Table 2. An example of computing descriptors C, T and D

Composition (C)	**A** 8/20	**B** 8/20	**C** 4/20		
Transition (T)	AB 6/19	BC 4/19	CA 3/19		
Distribution (D)	A_0%(1) 1/20	A_25% 2/20	A_50% 8/20	A_75% 12/20	A_100% 18/20
	B_0%(1) 3/20	B_25% 4/20	B_50% 9/20	B_75% 15/20	B_100% 19/20
	C_0%(1) 5/20	C_25% 5/20	C_50% 10/20	C_75% 17/20	C_100% 20/20

In this study, we adopted six global features derived from the physical or chemical characteristics (attributes) of proteins for fold classification. They are amino acid composition (C), predicted secondary structure (S), hydrophobicity (H), normalized Van Der Waals volume (V), polarity (P), and polarizability (Z). Among them, the first feature is simply the composition of amino acid sequence with its dimension equal to 20 whereas the remaining five features are extracted from the original primary sequence using the descriptor described above. The six kinds of protein sequence information (PSI) and their abbreviated symbol and dimension are shown in Table 3.

Table 3. The global features extracted from protein sequence.

Feature & its abbreviation	Characteristics used in descriptors			Feature Size
Composition (C)	20 kinds of amino acids			20
Predicted Secondary Structure (S)	Alpha	Beta	Loop	21
Hydrophobicity (H)	Positive	Neural	Negative	21
Volume (V)	Large	Middle	Small	21
Polarity (P)	Positive	Neural	Negative	21
Polarizability (Z)	Strong	Middle	Weak	21

2.2.2 Local Features

The six types of PSI introduced above are kinds of global features extracted by the direct encoding method. These six types of PSIs emphasize more on the global properties and structures of the amino-acid sequences, and less on the local interactions among neighboring amino acids. In this section, we will introduce two additional local features obtained by using the bi-gram coding method and the spaced-bi-gram coding method. These two induced features, which generated from the amino acid sequences, can well describe the interactions among neighboring amino acids locally in a 3D structure and also those caused by the mutual interactions among interleaving (every other) neighboring amino acids in a protein sequence. And it is quite reasonable because the protein sequence of amino acid exist in space instead of in line. There exists a force to influence others and the force is determined by different kind of amino acid. Based on this idea, the bi-gram and spaced-bi-gram coded features were used in our experiments.

Here, we would like to introduce the bi-gram coding method first whereas the spaced-bi-gram coding method will be introduced latter. For a given sequence composed of M

alphabets, an N-gram coding scheme applied on it will produce a feature vector with M^n dimensions and n is the N-gram size (length of window). Each element in the feature vector represents the number of appearance of a specific pair-wise combination of the M alphabets in the neighboring two alphabets. Since a protein sequence is composed of twenty kinds of general amino acids, these twenty kinds of amino acids are represented by twenty alphabets and others may be represented by a common alphabet B or Z; therefore it is a sequence composed of twenty-one alphabets (twenty kinds of amino acids plus an extra alphabet). In our experiments we took the windows length n as 2 and therefore the N-gram coding method becomes bi-gram coding method (or called 2-gram coding method) and we will obtain a feature vector with 441 dimensions for a protein sequence.

Similar to the bi-gram coding, the spaced-bi-gram coding is to detect the appearance frequency of any two-alphabet pair in every other (interleaving) neighboring amino acids of a protein sequence. This coding method we induced here is based on the concept of entropy and indirectly coding algorithm, we call it the spaced-bi-gram method. The spaced-bi-gram method will focus on the relationship between two neighbors of amino acids and count the transition of whole sequence to obtain information from the amino acid sequence, but with a space. In this manner, the transitions of the neighbor of amino acids sequence, but over one space, have been calculated. We believe that the structures of protein are not only influenced by the composition amino acids that combined with chemical bonds but also by other amino acids which do not connect each other directly but they are nearby. The concept is similar with the Van Der Waals force.

Same as the N-gram coding method, for an M alphabets sequence with windows length n, the new coding method will produce the pairs with the number of M^n. With the same reason we took the windows length equal to two. Hence, the spaced-bi-gram coding method on a protein sequence will also generate a feature vector with 441 dimensions. Let us consider a segment of the amino acid sequence of the protein with ID number 1pga as an example. If there is a segment occurred as ….MTYKLILNG….. in a sequence, we will use it as an illustrative example. In the bi-gram coding method, we count the numbers of the amino acid pairs (MT)、(TY)、(YK)、(KL)、(LI)、(IL)、(LN) … , respectively. But in the spaced-bi-gram coding method, for the same segment, we are going to count the numbers of the pairs (MY)、(TK)、(YL)、(KI) 、(LL)、(IN)、(LG)…, respectively. It is clearly to find that we consider the pairs jumping over an amino acid. It is believed that the mutual interactions between every two neighboring amino acids, and also the mutual interactions between every other two neighboring amino acids may play important roles in the space structure of a protein sequence. It is quite clearly that each amino acid with its volume, after tangle up in space it is not only affected by its neighbor but also affected by other composition.

The local features obtained by bi-gram and spaced-bi-gram here and also the six protein sequence information (PSI) in the previous section are listed in the Table 4.

Table 4. The global features and local features

Symbol	Protein Sequence Information	Dimension
C	Amino Acid Composition	20
S	Predict Secondary Structure	21
H	Hydrophobicity	21
P	Polarity	21
V	Normalized Van Der Waals volume	21
Z	Polarizability	21
B	Bi-Gram coding	441
SB	Spaced Bi-gram coding	441

2.2.3 Feature Sets of Combined Features

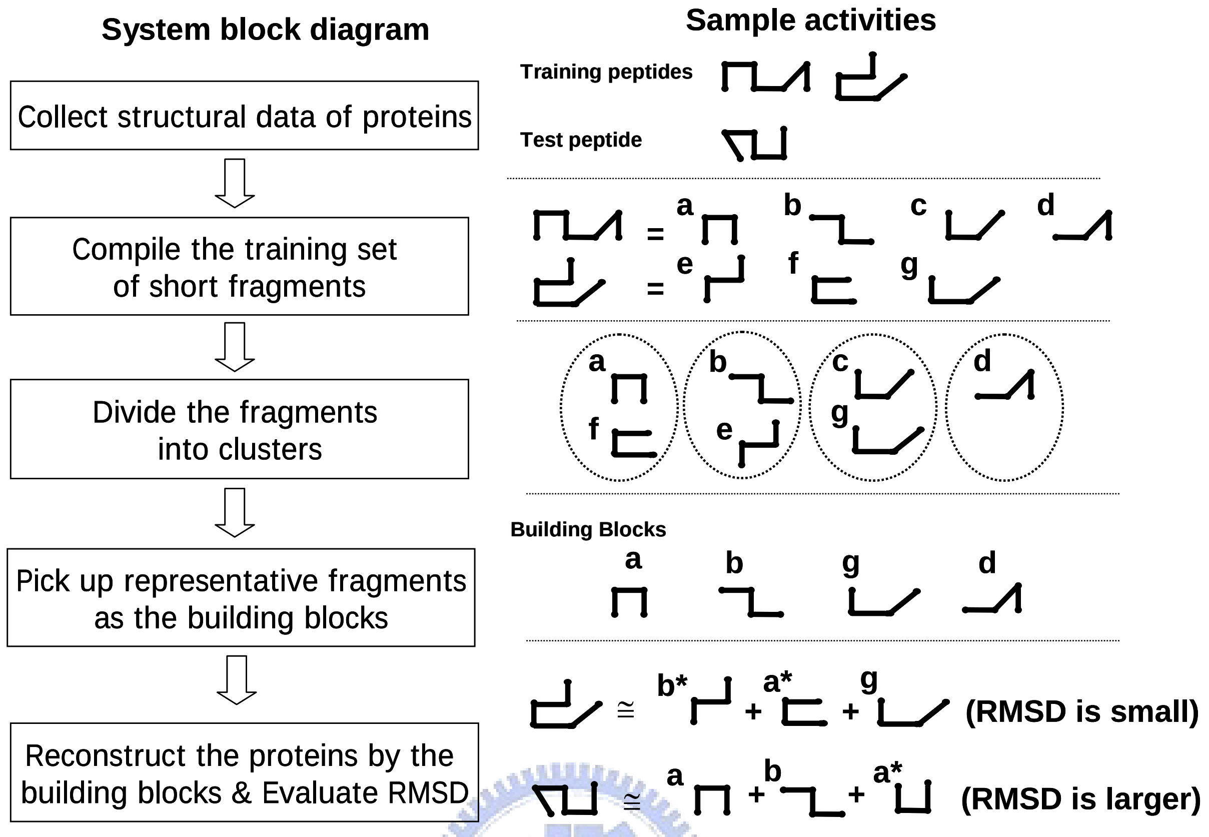
The global and local features are combined and used as input feature sets of classifiers. Table 5 gives the combined features sets.

Table 5. The combined feature set of original features.

Feature set	Features included	Dimension
A	C	20
B	C+S	41
C	C+S+H	62
D	C+S+H+P	83
E	C+S+H+P+V	104
F	C+S+H+P+V+Z	125
G	C+S+H+P+V+Z+B	566
H	C+S+H+P+V+Z+B+SB	1007

2.3 The 3-D Building Block Approach for Analysis of Protein Structure

3-D building blocks approach for analysis of protein structure is usually a multi-stage approach. First, we need to decide on the fragment length, i.e., the length of the building block. Then given a set of training proteins we generate fragments and use some clustering /data compression algorithm to divide these fragments into structurally similar clusters[11]-[14]. The center of each cluster is then used as the building block or prototype. Typically, the number of building blocks is much smaller than the total number of fragments. If there are an adequate number of good building blocks, then they can be used to represent the original fragments within a tolerable limit and hence in turn can be used to reconstruct the 3-D structure of a whole protein from its amino acid sequences within some tolerance.



System block diagram
Collect structural data of proteins
Compile the training set of short fragments
Divide the fragments into clusters
Pick up representative fragments as the building blocks
Reconstruct the proteins by the building blocks & Evaluate RMSD
Sample activities
Training peptides
Test peptide
a
b
c
d
e
f
g
Building Blocks
b*
a*
(RMSD is small)
(RMSD is larger)

Figure 1. A block diagram of the 3D building block approach.
(Note that fragment x* represents the fragment x after some rotation and alignment.)

For an easy understanding of the entire process, in the left part of Figure 1, we provide a block diagram and in the right side of each block we illustrate the activity in the block using a simple data set consisting of two training peptides and one test peptide. For illustration, we consider fragments of length 4. Thus, the two training peptides result in 7 fragments named **a-g** in Figure 1. In the third step (block), the clustering process finds four clusters. For example, in the first cluster, fragment **f** and fragment **a** are placed together because they are almost the same after alignment. The 4 clusters result in 4 building blocks, **a, b, g** and **d** as shown in the fourth block in the right part of Figure 1. In the last step we show two reconstruction cases of which the first one, (**b***+**a***+**g**), has a smaller reconstruction error (in terms of Root-Mean-Square-Deviation (RMSD)) than the second one since its corresponding building blocks are better matched. It is noted that **b*** represents the building block **b** after alignment.

2.4 Datasets used in the Dissertation

In this section, all the datasets used in this dissertation are listed below.

2.4.1 Dataset used for Protein Fold Prediction

2.4.1.1 Training Dataset

Following the prior published papers [5],[8],[21], the numbers of proteins for training are 313 and they should be classified into four main classes, i.e. all alpha(α), all beta(β), alpha and beta (α+β) and alpha/beta (α/β), with 27 folds according to their structures representing all major structural classes. To make sure the network will be well trained, the data set was selected by their characteristics so that all proteins in the data set have less than 35% of the sequence identity for the aligned subsequences longer than 80 residues. The protein list and corresponding 27 folds are shown in the Table 6 and 7.

Table 6. The twenty-seven folds used in the experiments of this study.

Globin-like	TIM beta/alpha-barrel
Cytochrome c	FAD/NAD(P)-binding domain
DNA/RNA-binding 3-helical bundle	Flavodoxin-like
Four-helical up-and-down bundle	NAD(P)-binding Rossmann-fold domains
4-helical cytokines	P-loop containing nucleoside triphosphate hydrolases
EF Hand-like	Thioredoxin fold
Immunoglobulin-like beta-sandwich	Ribonuclease H-like motif
Cupredoxin-like	alpha/beta-Hydrolases
Nucleoplasmin-like/VP	Periplasmic binding protein-like I
Concanavalin A-like lectins/glucanases	beta-Grasp
SH3-like barrel	Ferredoxin-like
OB-fold	Knottins
beta-Trefoil	
Trypsin-like serine proteases	
Lipocalins	

Table 7. Name list of training proteins used in the experiments of this study.

2lhb	3mdd	1ten	1cel	2hnt	1tph	1gky	1wht	1scy
3sdh	1bge	1cfb	1xnb	2kai	4xia	1ukz	1wht	1chl
1flp	1lki	3hhr	1lob	1bbr	1nfp	2ak3	2dri	1ktx
2hbg	1huw	3hhr	1shf	1hbq	2tmd	5p21	8abp	1ica
2mge	1gmf	2hft	1shg	1bbp	3cox	1tad	1gca	2gps
1eca	1rcb	3dpa	1pkt	1epa	1pbe	1eft	2liv	2cbh
2gdm	1hmc	1rsy	2hsp	1mup	1gal	1efg	1pda	1lpb
1bab	1ilk	1fru	1csk	1ifc	1trb	1dts	1sbp	1lpb
1ith	1ifa	8fab	1sem	1lid	2tpr	1nip	1olb	1tab
1ash	1rfb	1plc	1pse	1mdc	2tpr	2reb	1omp	
1hlb	4icb	1pmy	1sso	1cbs	3lad	2trx	1hsl	
1cpc	1rtp	1aaj	1lts	2phy	3lad	1mdk	1pbp	
1cpc	1cta	2aza	1bov	1cgt	1fcd	1dsb	1nnt	
1ccr	1rec	1cbp	1prt	6taa	1fcd	1ego	1pga	
1cxa	2scp	1aoz	1prt	1ppi	3chy	1aba	2ptl	
2pac	2sas	1aoz	1prt	1amg	1ntr	1gp1	1ubi	
2mta	2scm	1aoz	1kab	1amy	1scu	2gst	2pia	
1c53	6fab	1nrd	1tss	1byb	1scu	1gsr	1frd	
1fcd	1fc2	2bpa	1asz	1ghs	2fcr	1gsq	1put	
1fcd	3cd4	2bpa	1pyp	1xys	2fx2	3hsc	1tss	
1enh	1cid	2stv	1igp	1nar	4fxn	1atn	1fca	
1lfb	1hnf	4sbv	1csp	2ebn	1bmt	1atn	1fxr	
1apl	1dlh	2tbv	1bgh	1ctn	1cus	1glc	2atc	
1hdp	1dlh	2bbv	1rip	1add	2nad	1glc	1pba	
1hcr	1vaa	1bbt	1bar	2kau	1gdh	1chm	1nhk	
1ret	1vaa	1bbt	8i1b	1fba	2ohx	2rn2	1sxl	
1mse	1cd8	2cas	1ilr	1ads	1qor	1hrh	1nrc	
1mse	1tlk	4rhv	2ila	4enl	1hdc	1dpi	2bop	
1leb	1tnn	4rhv	1abr	2mnr	1dhr	1hjr	3rub	
3gap	1gof	4rhv	1abr	1chr	1udp	1ack	5rub	
1hst	1cgt	2mev	1tie	1oyb	1hdg	1ede	1aps	
1hks	1oxy	2mev	1hce	1gox	2nad	1tht	1ris	
1lpe	1clc	2mev	1arb	2tmd	1gdh	1tca	1efg	
1was	1ctn	1fod	2sga	1pii	2cmd	3tgl	4cpa	
256b	1nci	1len	2alp	1wsy	1bmd	1tib	1oma	
2ccy	2mcm	2ayh	1ept	1pkn	1lld	1crl	2sn3	
2hmz	1xso	1slt	1ppb	3rub	2pgd	1tah	1ptx	
2tmv	1fna	1sac	2snv	5rub	1scu	1lpb	2crd	

2.4.1.2 Testing dataset

The testing dataset was based on PDB-40D set developed by the authors of the SCOP database [1], [18]-[21]. A total number of 385 proteins with identity less 40%, same as those used by Ding and Dubchak [8], were selected for testing. Table 8 gives name list of the testing proteins. Table 9 shows the numbers of proteins in the training and testing datasets for different folds of each protein class used in our experiments, where there are 27 folds for the 4 main classes in total.

Table 8. Name list of testing proteins used in the experiments

2cmd_1	1dar_2	2btfa2	1pea	1mli	1vaoa1	1egf
1bdma1	1mmd_	2yhx_1	1pnra2	1pil	1geo_1	2tgf
1hlpa1	2	2yhx_2	1tlfa	1npk	1geo_2	1hre
1hyha1	1adea	1glag1	2lbp	1up1_1	1wgta1	1emo_1
1ldm_1	2reb_1	1glag2	1igd	1up1_2	1wgta4	1emo_2
1ldg_1	1bmfa3	1asu	1guab	1urna	1wgta2	1apq
1ldb_1	1bmfd3	1itg	1alo_2	2u1a	1wgta3	1klo_1
1yvei2	1kte	1bco_2	2pia_3	1dhma	1gur	1klo_3
2pgd_2	1mek	1kfd_1	1esfa2	1vhia	1iva	1klo_2
1hrda1	1dsba2	1noya	1se4_2	3rubl2	1eit	1flei
1gtma1	2gsta2	1sfe_2	1tif	5ruba2	1txm	1skz_1
1leha1	1glqa2	1wht.1	1lgr_1	1dar_4	1tsk	1skz_2
1deka	1gsea2	1din	1fd2	1afi	1pnh	
1vtk	1gnwa2	1broa	1xer	1psda3	1gps	
1aky	2trcp	1thg	1vjw	1mla_2	1cbh	
1tada2	1hpm_1	1tahb	2fxb	1fwp	1esl_2	
1hura	1hpm_2	1hpla2	1raab1	1regx	1hcgb	
1eft_3	2btfa1	1yasa	1spbp	1ab8a	1prha2	

Table 8. (cont.) Name list of testing proteins used in the experiments.

1hbg	1cpq	1wiu	1kcw_4	3ulla	1cbg	1gal_1
1mbd	2hmqa	1tiu	1kcw_5	1jmca1	1hvq	1gnd_1
1mba	1vtmp	1gof_1	1kcw_6	1jmca2	1edt	1gesa1
1lh1	1buca1	1qba_1	1stma	1mjc	1ctn_2	1gesa2
1baba	1fapb	1svb_1	1smva	1sro	1qba_3	1tde_1
1alla	1bgc	1cdg_1	1bmv1	1ah9	1pta	1tde_2
1dvh	1cnt1	1lla_2	1bmv2	1ckma1	1nal1	1nhp_1
1cyi	1csga	1hc2_2	1cwpa	1yhb	1dhpa	1nhp_2
5cytr	2int	1clc_2	2bbva	1pfsa	1ucwa	1lvl_1
1cc5	1hula	1ctn_1	1bbt3	1gpc	1dosa	1lvl_2
351c	1hmca	1ggta1	1pvc1	2prd	2acr	1rnl_2
1gks	3inkc	1ksr	1pvc2	2fgf	1ak5	1srra
1aofa1	1jli	1rhoa	1pvc3	1i1b	1ebha1	1scub1
1etpa1	1rmi	2hft_1	1tme1	1wba	2mnr_1	1rcf
1etpa2	1sra	2hft_2	1sva1	1hcd	2chr_1	5nul
1yrna	1rro	1cfb_1	1dhx	1sgc	1dora	1qrda
1yrnb	1osa	3hhrb1	1scs	1agja	1oya	1reqa2
1octc1	1scmb	3hhrb2	1cpn	1bty	1pii_1	1reqb2
1fjla	2mysb	1ebpa	1slaa	1hava	1nsj	1orda1
1res	1tcob	1cto	1lcl	1hbp	1pii_2	1esc
1pdnc	1djxa1	1bgla1	1kit_1	1obpa	1igs	2naca1
1igna1	1cpo_1	1bgla2	1kit_2	1beba	1pkya2	1dxy_1
1igna2	1cpo_2	1bhga1	1bia_2	1epba	1dik_1	1psda1
1sfe_1	1neu	1ggta2	1umua	1hms	3rubl1	1keva2
1bia_1	3cd4_1	1ggta3	1ckaa	1lfo	5ruba1	1xel
1lea	1cid_1	1noa	1pht	1eal	1tpfa	1cyda
1aoy	1cdcb	1yaia	1hsq	1cdg_4	2xis	1fds
1cgpa1	1vfba	1mspa	1mmd_1	1ppi_2	1luca	1fmca
1opc	1cd1a1	4kbpa1	1vie	2aaa_2	1lucb	1eny
1etd	1dlha1	2cbp	1ihwa	1jdc_2	1qapa1	1ybva
1puee	1vcaa1	1aiza	1sty	1amy_2	1djxa3	1gd1o1
2hts	1zxq_1	1cur	1tiid	1xyza	1gym	1dapa1
1dpra1	3cd4_2	1jer	1prtb1	1edg	1reqa1	1dih_1
1xgsa1	1cid_2	1cyx	1esfa1	1cec	1reqb1	1ofga1
1fow	1vcaa2	1occb1	1se4_1	1ecea	1pud	1dpga1
2liga	1zxq_2	1kcw_1	1asya1	1ghr	1sfta2	2naca2
1bbha	2ncm	1kcw_2	1lyla1	1bgla5	1coy_1	1dxy_2
1cgo	1tnm	1kcw_3	1cuk_3	1bhga3	1pbe_1	1psda2

Table 9. Fold numbers of each class and pattern numbers of each fold in SCOP which was picked up to be training and testing patterns in this study.

Classes	Fold number per class (Training patterns per fold)		Fold number per class (Testing patterns per fold)	
All Alpha	6	13,7,12,7,9,7	6	6,9,20,8,9,9
All Beta	9	30,9,16,7,8,13,8,9,9	9	44,12,13,6,8,19,4,4,7
Alpha/Beta	9	29,11,11,13,10,9,10,11,11	9	48,12,13,27,12,8,14,7,4
Alpha+Beta	3	7,13,14	3	8,27,27
Total Number	27		27	

2.4.2 Dataset used for finding 3-D building blocks of protein structures

Two datasets are used for finding 3-D building blocks of protein structures and we call them as Dataset A and Dataset B. Dataset A consists of the same set of 82 proteins as used in Unger et al. [12]. Dataset B is the same as used by Kolodny et al. [14] excluding a few proteins with sequence discontinuity. When creating the library of short fragments, only the c_α coordinates are used.

2.4.2.1 Dataset A

Dataset A is referred to as the "refined Brookhaven" database in [12]. Actually, Dataset A has two versions, Dataset A_{OLD} and A_{NEW}. The Dataset A_{OLD} is exactly the same database as used in [12]. The data in the Protein Data Bank are updated continuously as more new experimental observations become available. The Dataset A_{NEW} contains the same set of proteins as that in Dataset A_{OLD} but with updated information.

Table 10. Dataset A_{OLD}: Refined Brookhaven Peptides

1APR	1BP2	1CC5	1CCR	1CPP	1CPV	1CRN	1CTF
1ECA	1FB4h	1FBJl	1FC2d	1FDX	1GAPa	1GCR	1HIP
1HMQa	1INSa	2INSb	1LHl	1LZl	1LZT	1MBD	1NXB
1PCY	1PP2r	1PPD	1PFT	1SBT	1SN3	1TGSi	2ABXa
2ACT	2ALP	2APP	2AZAa	2CAB	2CCYa	2CDV	2CTS
2CYP	2ESTe	2FD1	2GN5	2INSa	2LHB	2LZM	2OVO
2PABa	2PKAa	2PKAb	2RHE	2SGA	2SNS	2SODo	351C
3C2C	3DFR	3ICB	3PGM	3PTP	3RP2a	3RXN	3SGBe
3TLN	4ADH	4APE	4ATCa	4ATCb	4CYTr	4DFRa	4FXN
4HHBb	4HHBc	4HHBd	4SBVa	5CPA	5LDH	5PTI	5RSA
5RXN	7CAT						

Table 11. Updated peptides list of Dataset A_{NEW} with new PDB number in the parentheses

1APR(2APR)	1GCR(4GCR)	3PTP(5PTP)	1GAPa(1G6Na)
1CPP(2CPP)	1HMQa(2HMQa)	3RXN(7RXN)	2FD1(5FD1)
1CPV(5CPV)	1INSa(4INSa)	3TLN(8TLN)	4FXN(2FOX)
1FB4h(2FB4h)	1PCY(1PLC)	4ADH(8ADH)	2APP(3APP)
1FBJl(2FBJl)	1SN3(2SN3)	4ATCa(6AT1a)	4CYTr(5CYTr)
1FDX(1DUR)			

Table 11 displays the list of updated peptides in Table 10 that have changed over time. The new PDB numbers are indicated in parentheses. We have used both Dataset A_{OLD} and A_{NEW} to evaluate the performance of our algorithms. Unger et al. [12] used four proteins (1BP2, 1PCY, 4HHBb, 5PTI) as the training data and the remaining 78 proteins as the test data; we also use the same protocols.

2.4.2.2 Dataset B

Table 12 and Table 13 include the list of proteins in Dataset B. The training dataset (Table 12) has 153 peptides whereas the test dataset (Table 13) has 144 peptides.

Table 12. Training data of Dataset B

1A1Ia	1A44	1A6M	1AAC	1ABA	1AH7	1AHO	1AIE
1AJSa	1AKO	1AMM	1AOHa	1ATZa	1AY7b	1B0Ya	1B2Pa
1B3Aa	1B4Va	1B67a	1BFD	1BFG	1BGF	1BKRa	1BM8
1BSMa	1BURs	1BX4a	1BXOa	1BYI	1BYQa	1C1Ka	1CBN
1CEX	1CIPa	1CJCa	1CSH	1CTF	1CY5a	1CYO	1CZFa
1D3Va	1D4Oa	1D7Pm	1DCIa	1DGFa	1DHN	1DMR	1DOZa
1DPSa	1DPTa	1EZM	1FND	1HFEl	1HFEs	1IFC	1IIBa
1IXH	1JHGa	1KAPp	1KID	1KPF	1KPTa	1KRN	1LAM
1LKKa	1MFMa	1MLA	1MOQ	1MRJ	1MSI	1MTYg	1MUN
1NKD	1NLS	1NOX	1ORC	1PCFa	1PDO	1PHC	1PHP
1PIDa	1PIDb	1POA	1PPN	1PTF	1QAUa	1QDDa	1QGUa
1QGXa	1QH4a	1QH5a	1QHFa	1QHVa	1QIPa	1QJ4a	1QKSa
1QQ5a	1QREa	1QS1a	1QSAa	1QTSa	1RA9	1RB9	1RGEa
1RIE	1RZL	1SGPi	1SMD	1SWUa	1T1Da	1TFE	1THW
1TTBa	1TX4a	1UBPa	1UTEa	1UTG	1VCC	1VFYa	1VHH
1VNS	1WHI	1YGE	1YVEi	256Ba	2BBKl	2CPGa	2CPL
2END	2ERL	2FDN	2GSTa	2IGD	2ILK	2LISa	2PTH
3BTOa	3CHBd	3CYR	3EBX	3EUGa	3EZMa	3GRS	3LZT
3PTE	3PYP	3SIL	3VUB	5PTI	7A3Ha	7ATJa	7RSA
8ABP							

Table 13. Testing data of Dataset B

1ABP	1ACX	1BDS	1BMV1	1BMV2	1BP2	1CC5	1CD4
1CHOi	1CLA	1CMS	1COHb	1CRN	1CSC	1CSEe	1CTS
1CY3	1ECA	1EST	1F19h	1F19l	1FC1a	1FC2c	1FDX
1FX1	1GRCa	1HIP	1HOE	1L12	1LH1	1LH4	1LZ1
1MBA	1MBD	1OVOa	1P09a	1PAZ	1PCY	1PFKa	1PHH
1PP2r	1PPT	1PRCc	1PRCh	1PRCl	1PRCm	1PYP	1RBBa
1REI	1RHD	1RMU1	1RNT	1SGT	1TECi	1TIM	1TNFa
1WRPr	1WSYb	256Ba	2AAT	2ACT	2ALP	2AT1b	2AZA
2CAB	2CCYa	2CDV	2CNA	2CPP	2CYP	2DHFa	2FD2
2GBP	2GD1o	2GLSa	2GN5	2HHBa	2HHBb	2HLAa	2HLAb
2I1B	2KAIb	2LIV	2LZM	2MEV1	2MEV3	2PABa	2PCY
2PKAa	2PKAb	2R063	2RSPa	2SBT	2SGA	2SNS	2SODb
2SSI	2STV	2TAA	2TAAa	2TBVa	2TMVp	2UTGa	2YPIa
351C	3ADK	3B5C	3BLM	3CA2	3DFR	3FXC	3GAPa
3GPDr	3GRS	3HMGa	3HMGb	3ICD	3PGK	3PGM	4AIT
4APE	4DFRa	4ER4e	4HVPa	4MDHa	4SBVa	4SGBi	4TLN
4TMNe	4TS1a	5CPA	5CPV	5EBX	5LDH	5MBN	5TNC
5XIAa	6ACN	7CATa	8ADH	8APIa	8APIb	8CATa	9PAP

3. Protein Fold Prediction by Data Fusion approach

3.1 Introduction

High technology large-scale sequencing projects have produced a massive number of proteins with putative amino acid sequences but much less is known in terms of their three dimensional (3-D) structure. Several popular structure databases, such as the Structural Classification of Proteins (SCOP) [18] and the Class, Architecture, Topology, and Homologous superfamily (CATH) [22], contribute only no more than 32000 entries in the Protein Data Bank (PDB) (SCOP release version 1.65 [7]: 20619 PDB entries, PDB: 31217 entries in 07-Jun. 2005). This number constitutes only about 20% of collections in the Swiss-Port (Swiss-Port release version 47.2: 184304 entries in 07-Jun. 2005). Physically, *x*-ray diffraction or NMR is used to determine the 3-D structure for a protein. However, each has its limitation [5]. As such, extracting structural information from the sequence databases becomes an important and complementary alternative, especially for swiftly determining protein functions or discovering new compounds for medical or therapeutic purposes.

The classification of protein structures has, more recently, been facilitated with some computer-aided algorithms. Previous research [5]-[6] have shown that an accuracy rate of 70-80% has been achieved to classify most of proteins into four classes according to their amino acid sequence information (i.e., all-alpha (α), all-beta (β), alpha/beta (α/β) and alpha+beta ($\alpha+\beta$)) [1]. In summary, these four classes contain 82.5% folding patterns, 84.7% superfamilies and 88.1% families in the SCOP database (SCOP release version 1.65 [7]). However, less optimal results are obtained if a more complicated category is used such as the one with protein folding patterns in [8].

In [8], Ding and Dubchak proposed a taxonmetric approach for protein folding classification (into 27 folding patterns) beyond four simple classes with a Neural Network (NN) and Support Vector Machine (SVM) [23]. Their approach attempts to predict the 3-D structure of a protein from its primary amino acid sequence under the assumption that only limited folding patterns are formed in most of these four protein classes and can be used as 'template'. They predicted protein folds according to six single-parameter features 'C', 'S', 'H', 'P', 'V' and 'Z' (see Section 2.2 for detail) first, then a combinatorial multiple-parameter features were formed and checked for their prediction accuracy in protein folding classification. They then demonstrated that one multiple-parameter feature 'CSHP' had the highest overall prediction accuracy rate at 56.5% by SVM.

In Huang et al. [9], extra features were defined. They proposed two additional indirect coding features 'B' and 'SB' (see Sections 2.2 and 2.3 for detail) to correlate 'neighboring' di-peptide pairs with protein structure classification. In addition to NN and SVM, they also constructed a new computational architecture called hierarchical learning architecture (HLA). In HLA, which was the first two-level classification strategy, a protein is classified into one of four classes at first, and then further classified into a folding structure (into one of 27 folding patterns). They combined the six single-parameter features proposed by Ding and Dubchak [8] and the outcomes of the two indirect coding features to form two new multiple-parameter features 'CSHPVZ+B' and 'CSHPVZ+B+SB'. With the latter features, Huang et al. [9] improved the prediction accuracy rate by 9%, compared with the result from Ding and Dubchak [8].

In this study, we apply the technique of data fusion [24]-[28], in particular the Combination Fusion Analysis described in Hsu et al. [27], to perform better protein structure classification, and better feature selection and combination. Using data fusion, results from various features are combined to obtain predictions with higher accuracy rate. In addition, the notion of **diversity rank/score function** is used to select the most suitable features for combination. We start with eight features, six from Ding and Dubchak (‘C’, ‘CS’, ‘CSH’, ‘CSHP’, ‘CSHPV’ and ‘CSHPVZ’) [8] and two from Huang et al. [9] (‘CSHPVZ+B’ and ‘CSHPVZ+B+SB’) to assign protein class and folding pattern. Then, some explicit rules from data fusion in information retrieval (IR) and virtual screening (VS) (see [24]-[28]) are used together with a special diversity rank/score graph to choose the best discriminating features for further combination. It has been demonstrated in IR and VS that using a combination of distinctive features may result in higher prediction accuracy rate than using single features. The proposed rules for proper feature selection are to reduce the complexity at the beginning. Then, we systematically choose the best discriminating features according to the diversity (see Section 3.2 for detail) of these features, which is represented in a diversity rank/score graph. Our experimental results achieves an overall prediction accuracy rate at 87% for predicting protein classes and 69.6% for predicting protein folding patterns which are higher than the previous work at 83.6% and 65.5% by Huang et al. [9], respectively.

3.2 Computational Framework and Architecture

3.2.1 Protein Datasets

We use the data sets from Ding and Dubchak [8] which were originated from the SCOP database for training and testing. Training data set is selected from the database built for the prediction of 128 folding patterns in the SCOP database [21]. It is ensured that any pair of two proteins in the training set is less than 35% identical in any aligned subsequence longer than 80 residues. The independent testing set is selected from the PDB-40D set [1], [18]-[21]. Moreover, all proteins in the testing set are less than 40% identical to each other. No protein in the testing set is more than 35% identical to any protein in the training set. The total number of proteins is 698 with 313 and 385 for training and testing, respectively. These proteins will be divided into 4 classes and 27 folding patterns all together according to their structures. Table 14 shows the number of proteins in different classes and folding patterns used for training and testing in this study.

Table 14. The variety in protein structures for training and testing

Classes	Folding patterns	Number of proteins (Training)	Number of proteins (Testing)
1. all-α	1. α_1: Globin-like	13	6
	2. α_2: Cytochrome *c*	7	9
	3. α_3: DNA-binding 3-helical bundle	12	20
	4. α_4: 4-helical up-and-down bundle	7	8
	5. α_5: 4-helical cytokines	9	9
	6. α_6: Alpha; EF-hand	7	9
2. all-β	7. β_1: Immunoglobulin-like β-sandwich	30	44
	8. β_2: Cupredoxins	9	12
	9. β_3: Viral coat and capsid proteins	16	13
	10. β_4: ConA-like lections/glucanases	7	6
	11. β_5: SH3-like barrel	8	8
	12. β_6: OB-fold	13	19
	13. β_7: Trefoil	8	4
	14. β_8: Trypsin-like serine proteases	9	4
	15. β_9: Lipocalins	9	7
3. α/β	16. $(\alpha/\beta)_1$: (TIM)-barrel	29	48
	17. $(\alpha/\beta)_2$: FAD (also NAD)-binding motif	11	12
	18. $(\alpha/\beta)_3$: Flavodoxin-like	11	13
	19. $(\alpha/\beta)_4$: NAD(P)-binding Rossmann-fold	13	27
	20. $(\alpha/\beta)_5$: P-loop containing nucleotide	10	12
	21. $(\alpha/\beta)_6$: Thioredoxin-like	9	8
	22. $(\alpha/\beta)_7$: Ribonuclease H-like motif	10	14
	23. $(\alpha/\beta)_8$: Hydrolases	11	7
	24. $(\alpha/\beta)_9$: Periplasmic binding protein-like	11	4
4. $\alpha+\beta$	25. $(\alpha+\beta)_1$: β-grasp	7	8
	26. $(\alpha+\beta)_2$: Ferredozin-like	13	27
	27. $(\alpha+\beta)_3$: Small inhibitors, toxins, lectins	12	27

3.2.2 Features

Features extraction from the data is critical for meaningful results before these features can be subjected to machine learning techniques. Different features may result in different classifications. Two major approaches including direct and indirect coding have been used to extract features from the data. The direct one contains a vector for each peptide residue in the chain that characterizes the position, sequence length and so on. In indirect coding, the vector is assigned for each sequence which is position and length independent [9]. Ding and Dubchak [8] proposed six direct coding features for protein structure classification. These single-parameter features are global descriptions of a peptide chain representing the proteins. These features are based on physical, chemical and structural properties of the constituent amino acids.

The six single-parameter features are amino acid composition (C), predicted secondary structure (S), hydrophobicity (H), normalized Van Der Waals volume (V), polarity (P) and polarizability (Z). The five multiple-parameter features, ‘CS’, ‘CSH’, ‘CSHP’, ‘CSHPV’ and ‘CSHPVZ’ were constructed to classify protein folding patterns. Ding and Dubchak [8] finally determined one multiple-parameter feature ‘CSHP’ with the highest overall accuracy rate for protein structure prediction with SVM. The above eleven single and multiple parameter features all emphasize more on the global properties and structures of amino acid sequences than on the local interactions among neighboring peptides.

In Huang et al. [9], they used the N-gram concept while extracting features from the amino acid sequence of proteins. Two other indirect coding features, generated from the bi-gram (B) and the spaced-bi-gram coding (SB) scheme, respectively, were proposed. These features reflect the local interactions among neighboring peptides within the 3-D structure of a protein. We combined the six single-parameter features proposed by Ding and Dubchak [8] and the outcomes of the two indirect coding features to form two new multiple-parameter features ‘CSHPVZ+B’ and ‘CSHPVZ+B+SB’. We showed that using the feature ‘CSHPVZ+B+SB’ together with NN outperformed all single- or multiple-parameter features used by Ding and Dubchak [8] in terms of prediction accuracy rate for protein structure classification.

In this study, we start with eight features, ‘C’, ‘CS’, ‘CSH’, ‘CSHP’, ‘CSHPV’, ‘CSHPVZ’, ‘CSHPVZ+B’ and ‘CSHPVZ+B+SB’ to assign protein classes or folding patterns. Then, we use the method of data fusion for feature selection and combination in order to improve classification accuracy.

3.2.3 The HLA Computational Architecture

The NNs have been commonly used in many machine learning and data mining applications, such as input-output mapping and bioinformatics [29], [30]. We use NN as a multi-class classifier to build hierarchical learning architecture (HLA) for the purpose of protein structure prediction. The Multilayer Perceptron (MLP) and the Radial Basis Function Network (RBFN) are two popular NN models. The RBFN is a three-layer network with Gaussian function that is suitable to be a classifier [31] since the weights of RBFN are measured and adjusted according to the distance of data. It was shown [9] that the overall prediction accuracy rate for protein structure classification using RBFN is better than that using MLP. Therefore, we adopted the RBFN model in this study where one hidden layer and nodes will be generated automatically. The hidden layer nodes show the coordinate of training sample clusters.

The HLA framework, proposed in Huang et al. [9], consists of a two-level procedure. In the first level, a protein is classified into one of four classes by a multi-class classifier (classifier 1 in Figure 2). Then, in the second level, it is further classified into one of f_i folding patterns by the corresponding multi-class classifier (f_1, f_2, f_3 and f_4 is equal to 6, 9, 9 and 3 in classifier 1, 2, 3, and 4 respectively in Figure 2).

In Huang et al. [9], it has been shown that the HLA framework is an effective learning structure which reduces the number of classifiers, avoids the voting scheme, and directly indicates the reliability or confidence of the result predicted. Our current study incorporates data fusion in HLA for the testing data set, as shown in Figure 2. For the training data set, HLA is used without data fusion. To predict which of four classes a protein belongs to with HLA, we use eight individual features to assign class to each protein in the testing data set at first. Then, we use the technique of data fusion to select the best feature and to combine results for the protein class discrimination. Finally, the protein class is predicted with the combined feature. For protein folding patterns associated with each protein class, the eight individual features are used once more to assign protein folding patterns to each protein in the class. Similarly, data fusion is applied again for feature selection and combination in order to improve the prediction of protein folding patterns

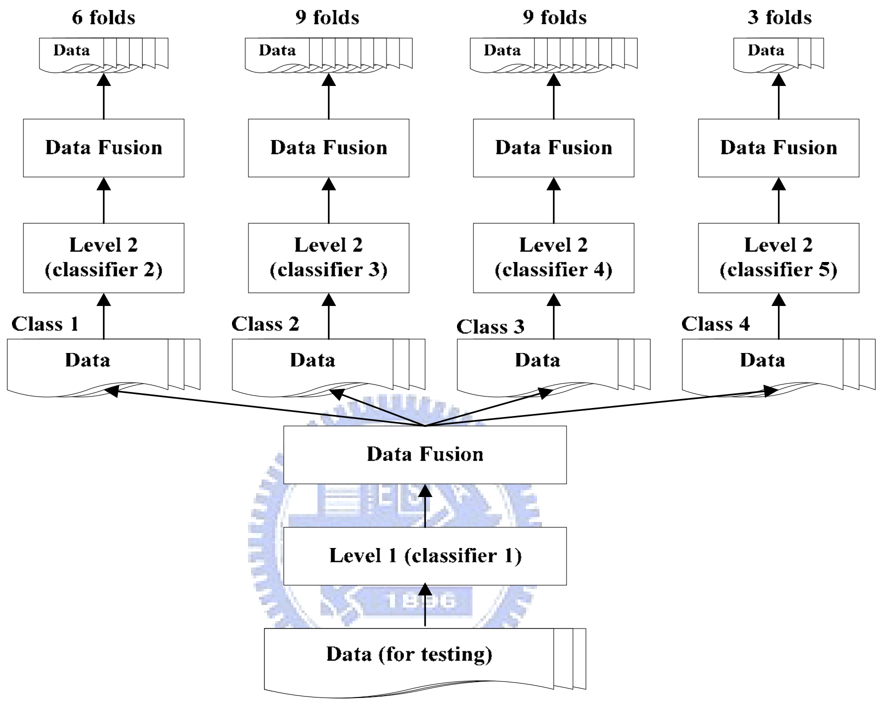


Figure 2. The architecture of HLA using data fusion

3.3 Data Fusion and Diversity Rank/Score Graph

The approach we take to properly select and combine features in protein structure classification is analogous to those used in information retrieval [24], [25], [28], [32], [33], pattern recognition [34], molecular similarity searching and structure-based screening [26], [35], and microarray gene expression analysis [36], [37], [38]. In addition, we adopt some of the notations and terminologies from [25], [26] and [27]. Moreover, each feature is viewed as a **scoring system** F containing a **score function** s_F and a **rank function** r_F on the set of classes.

Previous work in information retrieval, molecular similarity searching, structure-based virtual screening and microarray gene expression study have demonstrated the following:

Remark 1: For a set of multiple scoring systems, each with a score function and a rank function, we have

(a) the combination of multiple scoring systems would improve the prediction accuracy only if (1) each of the systems has a relatively high performance, and (2) the individual systems are distinctive (or diversified), and

(b) rank combination performs better than score combination under certain conditions.

Given a protein sequence and for each feature A, let $s_A(x)$ be a function that assign a real number to the class (or folding pattern) x in the set of all n classes (or folding patterns) $D = \{c_1, c_2, ..., c_n\}$. We view the function $s_A(x)$ as the score function from D to R (the set of real number) with respect to the feature A. When treating $s_A(x)$ as an array of real numbers, it would lead to a rank function $r_A(x)$ after sorting the $s_A(x)$ array into descending order and assigning a rank to each of their classes (folding patterns). The resulting rank function $r_A(x)$ is a function from D to $N = \{1, 2, ..., n\}$.

In order to properly compare and correctly combine score functions from multiple features, the function values have to be normalized. The normalization we used is the transformation from $s_A(x)$: $D \rightarrow R$ to $s_A^*(x)$: $D \rightarrow [0, 1]$ where $s_A^*(x) = \dfrac{s_A(x) - s_{\min}}{s_{\max} - s_{\min}}$, x in D and $s_{max} = max\ \{s_A(x) \mid x \text{ in } D\}$ and $s_{min} = min\ \{s_A(x) \mid x \text{ in } D\}$.

Suppose we have m features (i.e. m scoring functions). There are combinatorially, $2^m - 1$ combinations for all m individual features ($\sum_{k=1}^{m} \binom{m}{k} = 2^m - 1$) with rank or score functions. The total number of combinations to be considered for predicting protein class and protein folding pattern are $2^{m+1} - 2$ and $2^{2m+2} - 2^{m+3} + 4$ respectively in the HLA architecture. These numbers can become huge when the number of features *m* is large. Moreover, we have to evaluate the predictive power of each combination across all proteins. Because of this complexity, the current study would start with combining only two features which still retain fairly good prediction power. Combination of more than two features will be considered in our future work.

3.3.1 Methods of Combination and Feature Selection

Suppose *m* features A_i, $i = 1, 2, ..., m$, are given with score function s_{Ai} and rank function r_{Ai}, there are several different ways of combination. Among others, there are **score combination, rank combination, voting, linear average combination and weighted combination** [24]-[28], [32]-[39]. Voting is computationally simple and better than simple linear combinations when applied to the situation with large number of features. However, a better alternative is to reduce the number of features to a smaller number and then these features are combined. In this study, we reduce the set of features to those which perform relatively well and then use the diversity rank/score function to decide whether to combine by rank or by score. For the *m* features A_i, rank functions r_{Ai}, and score functions s_{Ai}, we have the score function s_R and s_S of the rank combination and score combination respectively defined as:

$$s_R(x) = \sum_{i=1}^{m}[(\mathrm{r_{Ai}}(\mathrm{x}))/\mathrm{m}], \text{ and } s_S(x) = \sum_{i=1}^{m}[(\mathrm{s_{Ai}}(\mathrm{x}))/\mathrm{m}]. \qquad (1)$$

As we did before, $s_R(x)$ and $s_S(x)$ are then sorted into ascending and descending order to obtain the rank function of the rank combination $r_R(x)$ and the score combination $r_S(x)$, respectively.

In this study, we use the rules (a)(1), (a)(2) and (b) stated in Remark as our guiding principle to select features and to decide on the method of combination. We started with eight features and, in each case, use rule (a)(1) to reduce the number of features to four. A diversity function $d(A,B)$ between features A and B is then defined using the concept of the rank/score function defined by Hsu et al [24], [25], [27].

3.3.2 Rank/Score Function and Diversity Rank/Score Graph

Given a protein sequence and for each feature A, we have the score function s_A and rank function r_A. Both s_A and r_A are functions from D to [0,1] and N respectively, where D = the set of classes. As in other application domains [24]-[27], we explore the scoring (and ranking) characteristics of feature A by calculating the **rank/score function**, $f_A : N \rightarrow [0, 1]$ as follows:

$$f_A(j) = (s_A^* \circ r_A^{-1})\,(j) = s_A^*\,(r_A^{-1}(j)). \qquad (2)$$

We note that the set N is different from the set D which is the set of classes (or fold patterns). The set N is used as the index set for the rank function value and $|N| = n$ is indeed the cardinality of D. The rank/score function so defined signifies the scoring (or ranking) behavior of the feature A and is independent of the classes (or folding patterns) under consideration.

For protein p_i in $P = \{p_1, p_2, ..., p_t\}$ and the pair of features A and B, **the diversity score function** $d_i(A,B)$ is defined as: $d_i(A,B) = \Sigma\left|f_A(j) - f_B(j)\right|$, where j is in $N = \{1, 2, . . ., n\}$ and n is the number of classes (or folding patterns). When there are q features selected (in this study, q = 4), there are $\binom{q}{2} = \frac{q(q-1)}{2}$ (in this study, this number is 6) diversity score functions. If we let i vary and fix the feature pair (A,B), then **$d_i(A,B)$ is the diversity score function $s_{(A,B)}(x)$ from $P = \{p_1, p_2, ..., p_t\}$ to R.** Sorting $s_{(A,B)}(x)$ into descending order would lead to **the diversity rank function $r_{(A,B)}(x)$.** Consequently, the **diversity rank/score function** $f_{(A,B)}(x)$ is defined as:

$$f_{(A,B)}(j) = (s_{(A,B)} \circ r_{(A,B)}^{-1})\,(j) = s_{(A,B)}\,(r_{(A,B)}^{-1}\,(j)), \text{ where } j \text{ is in } T = \{1, 2, 3, ..., t\}. \qquad (3)$$

We note that the set T is different from the set P which is the protein set considered. The set

T is used as the index set for the diversity rank function value and $|T| = t$ is indeed the cardinality of P. The diversity rank/score function $f_{(A,B)}(k)$ so defined exhibits the diversity trend of the feature pair (A,B) across the whole spectrum of input set of t proteins and is independent of the specific protein under study.

For two features A and B, the graph of the diversity rank/score function $f_{(A,B)}(j)$ is called the **diversity rank/score graph** (or **diversity graph** in short). Our current study aims to examine all the $\frac{q(q-1)}{2}$ diversity rank/score graphs to see which pair of features would give the highest diversity measurement. Following rules (a)(2) and (b) in Remark 1, the rank combination of these two features is then calculated to give the final rank function and to choose the class (or folding pattern).

3.4. Results

The technique of combinatorial fusion (see [27]) is used for protein structure classification on a testing data set with NN using RBFN under the HLA architecture. Initially, we use eight features, 'C' (reworded as A), 'CS' (as B), 'CSH' (as C), 'CSHP' (as D), 'CSHPV' (as E), 'CSHPVZ' (as F), 'CSHPVZ+B' (as G) and 'CSHPVZ+B+SB' (as H), to assign protein classes for all proteins tested. Following the rule (a)(1) in Section 3.1, we select four features E, F, G and H, for further fusion (or combination) because of their higher accuracy rate than others as demonstrated in [9]. With the help of rule (a)(1), we can reduce 2^8-1 combinations to 2^4-1 combinations. Following the rules (a)(2) and (b) in Section 3.1, we shall use the rank combination of the features to predict the protein class.

As stated in Section 3.2, the diversity of any two of features E, F, G and H can be calculated for all proteins tested and features E and H are found to have the highest diversity, as shown in Figure 3, among all six ($\binom{4}{2} = 6$) feature combinations. In conjunction with (b) in Remark 1, we use the rank combination of features E and H to predict protein classes for all proteins tested. After the protein classes for all proteins tested have been predicted and categorized, the prediction of protein folding patterns follows in the HLA architecture. We use the same rules and a diversity graph to choose the best combined two features in each class for the purpose. Accordingly, we choose a rank combination of features BG, GH, DH and GH to predict protein folding patterns in classes 1, 2, 3 and 4, respectively. The diversity graph to pick the pair of features (B,G), (G,H), (D,H) and (G,H) for combination and to predict folding patterns in class 1, 2, 3 and 4 are depicted in Figure 4(a),(b),(c) and (d), respectively. In Figures 4(b) and (d), only the pair of features (G,H) is selected since its accuracy rate is higher than others. It implies that the features G and H are more suitable than others for classifying proteins, which belong to class 2 or class 4, into folding patterns.

We use the standard percentage accuracy rate Q_i [8], [9], [40] to evaluate our work. $Q_i = p_i/n_i \times 100$, where n_i is the number of testing proteins in the ith class or folding pattern and p_i is the number of proteins being correctly predicted in the ith class or folding pattern. The overall prediction accuracy rate Q is given by $Q = \sum_{i=1}^{k} q_i Q_i$, where $q_i = n_i/K$, where K is the total number of proteins tested, and n is the number of classes or folding patterns. We compare the overall prediction accuracy rates Q for protein classes in the previous work [9] and current work. These are shown in Table 15. The current overall prediction accuracy rate is 87%, 3.4% higher than that of the previous work. Table 16 shows that for prediction of folding pattern, our current work has an overall prediction accuracy rate of 69.6%, which is 13.1% higher than that of Ding and Dubchak [8], 4.1% higher than that of the previous work.

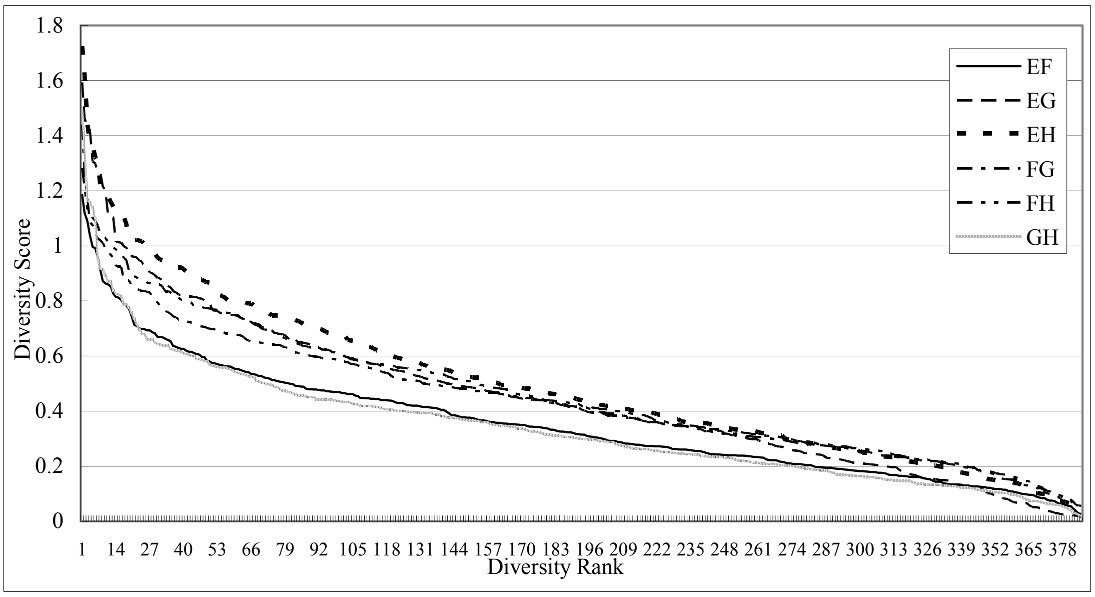
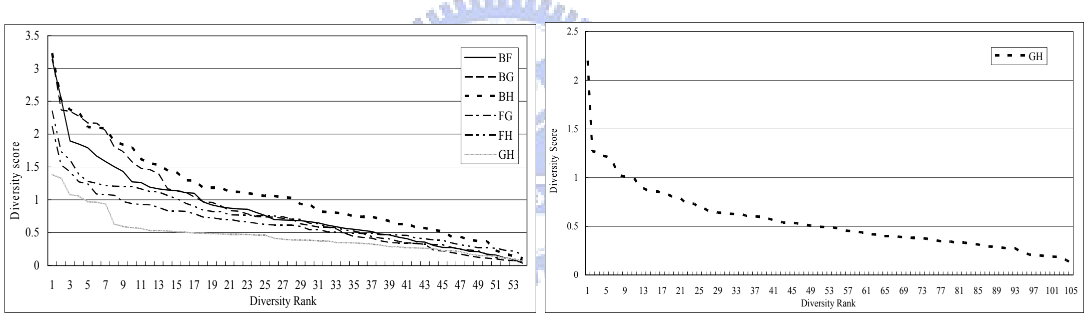
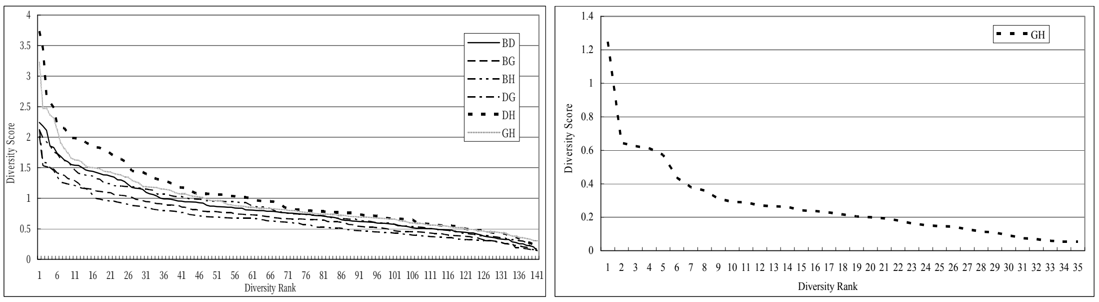


Figure 3. The diversity rank/score graph for each pair of features from {E,F,G,H}for classifying protein classes



(a) Class 1 (b) Class 2



(c) Class 3 (d) Class 4

Figure 4. The diversity rank/score graph for each pair of features in {B,F,G,H}for classifying protein folding patterns in class1; in {G,H}for classifying protein folding patterns in class2; in {B,D,G,H}for classifying protein folding patterns in class3; and in {G,H}for classifying protein folding patterns in class4

Table 15. The comparisons of overall prediction accuracy rates *Q* for protein classes

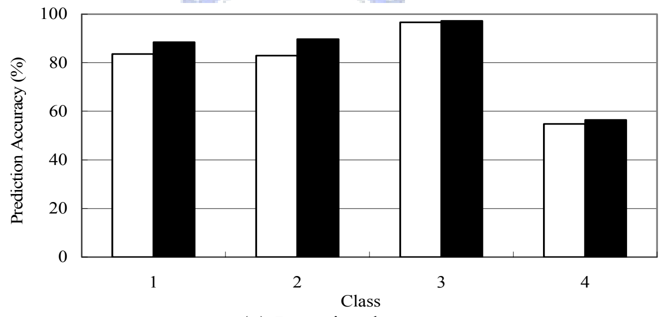
Method	HLA, NN 'CSHPVZ'*	HLA, NN 'B'*	HLA, NN 'CSHPVZ+B'*	HLA, NN 'CSHPVZ+B+SB'*	HLA + data fusion, NN
Q	81.6	79.2	83.1	83.6	**87**

* Data from Huang et al. [9]

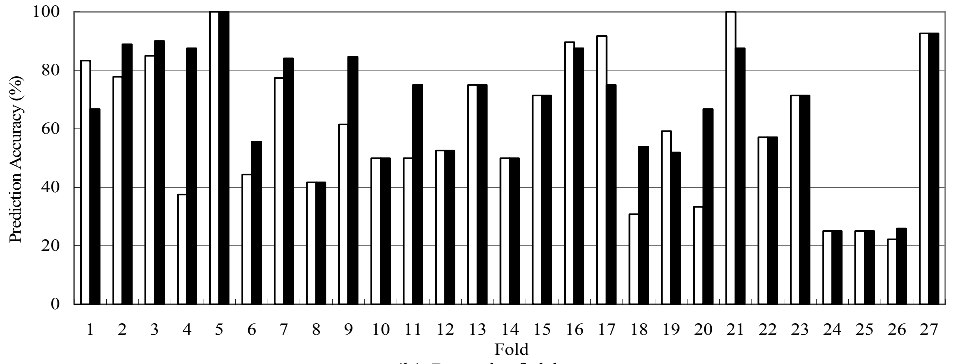
Table 16. The comparisons of overall prediction accuracy rates *Q* for protein folding patterns

Feature Method	'C'	'CS'	'CSH'	'CSHP'	'CSHPV'	'CSHPVZ'	'CSHPVZ+ B'	'CSHPVZ+ B+SB'
OvO[1], NN**	20.5	36.8	40.6	41.1	41.2	41.8	—	—
OvO[1], SVMs**	43.5	43.2	45.2	43.2	44.8	44.9	—	—
uOvO[2], SVMs**	49.4	48.6	51.1	49.4	50.9	49.6	—	—
AvA[3], SVMs**	44.9	52.1	56.0	**56.5**	55.5	53.9	—	—
HLA, NN*	44.9	53.8	53.3	54.3	55.3	56.4	63.7	**65.5**
HLA+data fusion, NN	**69.6**							

[1]one-versus-others method [8]; [2]unique one-versus-others method [8]; [3]all-versus-all method [8]
(* Data from Huang et al. [9]   ** Data from Ding and Dubchak [8])



(a) Protein classes



(b) Protein folds

Figure 5. The comparisons of prediction accuracy rates Q_i of the previous work (Huang et al. [9]) (in white) and the current work (in black) (a) for 4 protein classes and (b) for 27 protein folding patterns

We summarize the comparisons of prediction accuracy rates Q_i of the previous work [9] and our current work in Figure 5. Our results give prediction accuracy rates (>80%) in 3 classes, especially in class α/β with accuracy rate reaches 97.2%, all higher than what those achieved previously, shown in Figure 5(a). For protein folding patterns prediction, the current work gives prediction accuracy rates (>80%) in 9 folding patterns, more than what in the previous work[9], as shown in Figure 5(b). Also, the current work outperforms the previous work in 10 folding patterns, especially (> 30% improvement) in folding patterns: α_4 (4-helical up-and-down bundle), β_3 (Viral coat and capsid proteins), β_5 (SH3-like barrel), $(\alpha/\beta)_3$ (Flavodoxin-like) and $(\alpha/\beta)_5$ (P-loop containing nucleotide). The previous work has slightly better results only in 5 folding patterns (especially in fold α_1 (Globin-like)). Overall, there is an improvement with our current method using the HLA framework and data fusion techniques. In summary, the current method has achieved an accuracy rate of 69.6% for folding pattern classification, which is a significant improvement over the result of Ding and Dubchak ([8], 2001) of 56.5%.

3.5. Summary and Discussion

Methods of combining multiple classification systems or multiple scoring systems have been used in a variety of applications domains including information retrieval, pattern recognition, microarray gene expression analysis, and molecular similarity searching [24], [28], [32]-[39]. More recently, criteria to select the classification systems or scoring systems for combination and to decide ways to combine these systems have been discussed and studied [25]-[28], [39]. It has been demonstrated in Combinatorial Fusion Analysis (see [27] and its references) that (a) the combination of multiple systems (or features) would improve the performance only if (1) each of the individual systems (features or functions) has a relatively high performance, and (2) each individual systems are distinctive (or different), and (b) combination by rank outperform combination by score under certain conditions.

In this study, we use criterion (a)(1) to select features and then apply criterion (a)(2) by computing the diversity rank/score graph in order to select the pair of features with the highest diversity. Criterion (b) is then used to combine these two features using ranks. We have applied the concept of Combinatorial Fusion to improve accuracy in protein structure prediction. In particular, we have successfully improved the overall predictive accuracy rate of 87% for the second structure (the four classes) and 69.6% for the folding patterns (the 27 folding categories). We improve previous results by Huang et al. [9] (65.5% for folding structure) and Ding and Dubchak [8] (56.5% for folding structure) by incorporating the method of combinatorial fusion in their approach using neural network (NN) with the radial basis function network (RBFN) using the hierarchical learning architecture (HLA).

One of the novelties of our current work is the notion of a diversity rank/score function $d_i(A,B)$ between a pair of features A and B (See e.g. Figures 3 and 4). This function characterizes the diversity of ranking (or scoring) behavior between features A and B across the whole spectrum of all protein sequences under consideration. This parameter is then used to select appropriate and diverse features for combination. The current work is the first of a series of on-going projects towards the protein structure prediction problem using HLA, NN-RBFN, and Combination Fusion Analysis. Following the current work, we have observed the following:

(A) The method of combinatorial fusion we used in this study is computational efficient, able to adapt to different situations and approaches, and scalable to a large number of classes (or folding patterns) and a large number of proteins.

(B) In this study, we considered only combination of a pair of two features in order to improve the performance. It may be possible to achieve better results with combination of more than two features. However, it is indicated in criteria (a)(1) and (2) that each of these three or more features would have relatively high performance and individual features should be different. As such, the diversity between three or more features should be defined. This will be studied in a latter work.

(C) Although it has been shown (e.g. [41]) that combining multiple predictors or servers improves fold recognition, we note here that combining all the features or multiple scoring systems together may not guarantee optimal performance (see also [26] and [27]).

(D) We used rank combination due to criterion (b) which was demonstrated to be better under certain conditions analytically and by simulation in Hsu and Taksa [25]. We observed that score combination does have its merit when the two features combined are similar and homogeneous with respect to their scoring functions, rank function, or rank/score function. We decided to use the rank combination because the pair of features to be combined satisfies criteria (a)(1) and (a)(2) and these two items are precisely the conditions stipulated in [25], [26], [28], [37] and [38].

(E) In our feature selection process, we selected top four performers out of the original eight features. The ideal case is to select those features which perform much better than the others. That means there is a big difference on the performance between those selected and those not selected.

Our current work represents the first of a series of investigations on the protein structure prediction problem using HLA and Combinatorial Fusion. It has generated several issues and topics worthy of further study. We summarize some of them here:

(1) Our diversity rank/score function $d_i(A,B)$ for the feature pair (A,B) with respect to protein p_i is defined using the variation of the rank/score functions between A and B. As indicated in [25], [27] and [28], variation of the rank functions or the score functions between A and B can be used also to define the diversity score function. We will explore these two other options in a latter work.

(2) The effectiveness of our fusion of multiple features is limited by the set of eight original chosen features. It might be worthwhile to study the content of original set of features. For example, we like to explore the diversity among the original features such as local vs. global, physical vs. chemical and bi-gram vs. tri-gram scheme.

(3) Related to observation (D) above, one might ask if it is better to expand the scope and the number of features. In this study, we started with eight features and four are selected using the CFA criteria. In a separate paper [42], eleven features are collected and three features are selected according to the criteria (a)(1) and (a)(2) in Remark 1. We have, in Lin et al. [42], obtained a slightly better overall accuracy rate of 87.8% for four classes and 70.9% for 27 folding categories.

(4) Our results improve previous results by Huang et al. [9] and Ding and Dubchak [8] which used neural network with radial basis function in a hierarchical learning architecture. Work has been performed to improve those results which used other machine learning technique such as kernel method, SVM and genetic algorithm. For example, Yu et al. [43] has obtained good accuracy rate using SVM with *n*-peptide coding schemes and jury voting. Future work can be performed to improve these results using our combinatorial fusion approach.

4. Finding 3-D building blocks of protein structures by Mountain Clustering Approach

4.1 Introduction

Discovering the relations between protein sequences and their 3-D structures is an important research topic and has received a lot of attention because knowing the 3-D structure of a protein helps biologists to study the functions of the proteins, perform rational drug design, and design novel proteins. Finding the 3-D structure of a protein using X-ray crystallography or by nuclear magnetic resonance imaging is time consuming and expensive and hence alternative approaches are being tried. Several approaches such as comparative modeling, fold recognition [9], [44], ab-initio prediction [45]-[46] and 3-D building blocks approach [12]-[13] have been proposed. As pointed out by Bujnicki [10] modeling of a protein structure de novo without using templates is quite difficult because the search space is enormous even for proteins with moderate sequence lengths. The methods based on assembly of short fragments have shown a great promise [10]-[14]. Among these methods, 3-D building blocks approaches have been successfully applied to construct libraries of well-chosen short structural motifs extracted from known structures [13]-[14], [47]-[54]. These building blocks are then used to construct or analyze structures of new proteins. The short structural fragments that recur across different protein families can often be viewed as stand-alone units which fold independently and hence can help assignment of building blocks to unknown proteins for reconstruction of 3-D structure [11]. The clustering method used in [11] is a two stage process, where building blocks are classified according to their SCOP protein family and clustered within the family in the first stage, and then merged in the second stage. The building block cutting algorithm uses a stability score function that involves properties like compactness, hydrophobicity, and isolatedness. The critical building block finding algorithm uses a score function based on the contacts the building block has with other building blocks. This is an involved and interesting approach. Our proposed approach is comparatively very simple and does not use those physical/ chemical/ structural properties of the residues.

In [52] Anishetty et al. suggested that rigid tri-peptides have no correlation with protein's secondary structure and tri-peptide data may be used to predict plausible structures for oligopeptides. The hybrid protein model of de Brevern et al learns 3-D protein fragments encoded into a structural alphabet consisting of 16 protein blocks (PBs) [54]-[55]. Benros et al. [53] further continued this study considering 11-residue fragments encoded as a series of seven protein blocks. They had built a library of 120 overlapping prototypes with good local approximation of 3-D structures. Every protein block in [54] is only five-residue long and described by eight dihedral angles. Each of them serves as a building block approximately representing a known structural motif like central α-helices, central β-strands, β-strand-N-caps and so on. Consequently, a protein's 3-D structure can be represented by a string of alphabets. And unlike our approach, the similarity between fragments is defined by the RMS deviation on angular values. The clustering algorithm used is a self-organizing map type neural network.

The effectiveness of such a method heavily depends on the extraction of good representative 3-D building blocks. Unger et al. [12] proposed a two-stage clustering algorithm to choose hexamers (fragments of length 6) having a large number of neighbors to be the building blocks. These center hexamers are called the 3-D building blocks [12]. Micheletti et al. also used largest number of nearby points within a similarity cutoff called "proximity score" [13] to select cluster centers, while Kolodny et al. proposed a simulated annealing k-means method to perform the clustering task with the minimal total variance score [14], [50]-[51]. In this study, a modified form of the mountain clustering / subtractive clustering method [15]-[16] is proposed to find building blocks. Our experiments with some benchmark datasets show that it can find better representative building blocks than the method in [12]. We also propose two alternative ways of depicting the quality of the building blocks.

4.2 3-D Building Block Approach

The 3-D building block approach involves several steps. First, we need to decide on the fragment length. Then, given a set of proteins (training data) we need to compile the whole set into all possible fragments of the selected length. Next, a clustering method is used to divide these fragments into clusters and pick up the center of each cluster to be a building block. If these building blocks are good enough, then they can be used successfully to represent all original fragments within a tolerable limit and therefore can be used to reconstruct the 3-D structure of a whole protein within some tolerance.

4.2.1 Distance Measure between 3-D structures

A well-accepted definition of dissimilarity between two fragments is the Root-Mean-Square (RMS) deviation between two structures computed after alignment of the two fragments to the greatest possible extent using the BMF (best molecular fit) algorithm [56], [57]. Given two structures s and t, the RMS can be calculated as follows:

$$RMS = [(\sum_{i=1}^{K} \left\| r_i^s - r_i^t \right\|^2)/(K-2)]^{1/2} \qquad (4)$$

where r_i^s is 3-D coordinate of i^{th} C_α atom in the molecule s and K denotes the number of atoms in the structure. Typically, for the computation of RMS, one should divide by K, but for the ease of comparison with published results, we divide by (K-2) as done in [12].

4.2.2 Method of Reconstruction

Following [12] we use this procedure: First, we replace each original hexamer of a protein by its closest building block. Then, since the building blocks overlap, we align every two consecutive building blocks using the BMF algorithm. The chain grows as follows. Onto the suffix (the last five residues) of the first building block we fit the prefix (the first five residues) of the next building block. The 3-D position of the sixth (last) residue of the latter hexamer is, thus, determined and is added to the growing chain. This process is repeated until the whole protein is reconstructed.

4.2.3 Performance Measure

To evaluate the performance of the proposed method, we use the same two criteria as in [14]: (1) Local-fit RMS, which measures how well the fragments of the target proteins can be represented by the library of building blocks at hand. It takes the average of all coordinate RMS deviations between every fragment and its associated building block. (2) Global-fit RMS, which measures the RMS deviation of the reconstructed 3-D structure from the entire native structure of the target. In addition we also use two alternative ways, as explained later, for assessment of quality of the building blocks.

4.3 Clustering Approach

4.3.1 Two Stage Clustering Algorithm (TSCA)

Since we shall compare our results with those by the algorithm in [12], we briefly describe the same. The TSCA defines a cluster as a set of structures such that the RMS deviation of any member in the cluster from a designated representative member is less than a threshold. In [12] 1 $\mathring{A}$ is used as the separation between similar and not similar hexamers, and hence as the threshold for defining clusters. In the first stage a randomly chosen hexamer is taken as the center of the first cluster and all hexamers which are within 1 $\mathring{A}$ distance after best molecular fit are placed in that cluster. Each member of this cluster then acts a new center and adds all of its neighbors which are within the threshold. This annexation process is continued till no more hexamer can be added to the cluster. Then another unused hexamer is taken as the center of the next cluster and the process is repeated to get the next cluster. The entire process is repeated till every hexamer is included in some cluster. It is obvious that in such a cluster the maximum distance between a pair of hexamers could be much higher than 1 $\mathring{A}$. In the second stage, these big clusters are divided into smaller clusters such that every member of a cluster is within a distance of 1 $\mathring{A}$ from a centroid. For each cluster, the hexamer with the maximum number of neighbors within 1 $\mathring{A}$ is taken as the center of a new sub-cluster having those neighbors as members. The process is repeated until all hexamers of the cluster are assigned to sub-clusters.

4.3.2 Mountain Clustering Method (MCM) and Subtractive Clustering Method (SCM)

Let $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\} \subset \Re^p$ be a set of n data points in p-dimension. We denote x_{jk} as the j^{th} component of the k^{th} point $\mathbf{x}_k$; $k = 1,2,...,n; j = 1,2,..., p$. The mountain clustering method [15] generates a set of N equispaced grid points $\mathbf{v}_i , i = 1,2,..., N$ in $\Re^p$ over the smallest hypercube (in $\Re^p$) containing X . Then at every grid point a potential value (called mountain potential) is computed which represents a kind of local density of points around the grid point. Now the grid point with the maximum mountain potential is selected as the first cluster center. To find other cluster centers, the mountain function values are "discounted" to reduce the effect of already detected centers and the grid point corresponding to the highest peak of the discounted potential is taken as the next cluster center. This process of discounting and finding of cluster center is continued until the discounted potential becomes too small to look for useful clusters.

In MCM the quality of the centers depends on the fineness of the grid and better resolution leads to more cost. The computational overhead increases rapidly with dimension p . To reduce the computational overhead of MCM Chiu [16] suggested a modification of MCM, known as the *Subtractive Clustering Method* (SCM).

Instead of imposing artificial grids, Chiu [16] suggested to use each *data point* as a potential cluster center. Following the MCM, the potential function is defined as:

$$P_1(\mathbf{x}_i)=\sum_{k=1}^{n} e^{-\alpha\, d^2(\mathbf{x}_k,\mathbf{x}_i)};\ i=1,2,\cdots,n; \tag{5}$$

and discounting the potential on subsequent steps is done as follows:

$$P_k(\mathbf{x}_i)=P_{k-1}(\mathbf{x}_i)-P_{k-1}^{*}\ e^{-\beta\, d^2\left(\mathbf{x}_{k-1}^{*},\mathbf{x}_i\right)} \\ ,\ k=2,\cdots,c;\ i=1,..,n \quad . \tag{6}$$

Here $\mathbf{x}_{k-1}^{*}$ is the $(k-1)^{th}$ (most recently detected) cluster center, and α and β are positive constants. The rest part of the SCM algorithm remains the same as that of mountain method. Unlike MCM, here the number of prospective cluster centers is n, and hence is not dependent on the dimensionality and spread of the data. Chiu [16] terminated SCM when $\frac{P_{k-1}^{*}}{P_1^{*}}<\delta,\ 0.0<\delta<1.0$. Although, SCM reduces the computational complexity, it will give good results only if the desired cluster centers (points corresponding to the maximum local density) coincide with one of the data points or close to it. For the present problem, since we have to choose one of the hexamers as the center, the SCM framework is quite suitable.

4.3.3 Structural Mountain Clustering Methods (SMCM)

This is a modified form of subtractive mountain clustering method [15]-[16] so that it can handle structural data such as hexamers. For hexamers, use of Euclidean distance will not be meaningful because the Euclidean distance between two hexamers where one is a translated version of the other or one is a rotated version of the other would be high, while for our purpose they are the same. Suppose the set of hexamers is represented by $X=\{\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_n\}\subset\Re^p$. In SMCM each hexamer is considered a potential cluster center. Instead of the Euclidean distance, the contribution made by a hexamer $\mathbf{x}_j$ to the potential associated with another hexamer $\mathbf{x}_i$;

$i \neq j$ depends on the *structural similarity* between $\mathbf{x}_j$ and $\mathbf{x}_i$. The structural similarity is obtained after aligning the data points using the BMF routine [56]-[57]. Thus the higher the similarity between two hexamers, the more quantity is added to the potential. In this way at every hexamer we compute the mountain potential P using all other hexamers. After this, like MCM we find the hexamer, $\mathbf{x}_k$, with the highest potential as the first building block. Now we form the first cluster taking all hexamers which are within 1 $\overset{\circ}{A}$ of RMS after best molecular fit. We now remove all members in the first cluster and *recompute* the potential to find the next cluster center. Note that, MCM and SCM neither remove any data point nor recompute the potential. Here we recompute the potential as we want every cluster center to be at the center of a dense region. To get the third cluster, the members of the second clusters are removed and the potential is recomputed. The process is continued until every data point is assigned to some cluster as described in the algorithm next:

Algorithm:

Input : Data set $X = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\} \subset \Re^p$

Choose : α

Compute $d(\mathbf{x}_i, \mathbf{x}_j) = RMS_{i,j}$, for all $i, j = 1,2,...,n$ using the BMF Algorithm; $RMS_{i,j}$ is the Root Mean Square distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ after BMF.

Repeat while any hexamer is left to be assigned to a cluster

1. Calculate the potential at each hexamer $\mathbf{x}_i$ using equation (5).

2. Find the hexamer with the highest potential and choose it as a building block.

3. Remove all hexamers, which are within a RMS error of 1 $\mathring{A}$ from the building block, to form the cluster associated with the building block.

End Repeat.

Choice of α may have effect on the clusters extracted and hence we experimented with different choices of α to get an optimal value for it.

4.3.4 SMCM Can Produce Better Building Blocks Than TSCA

Note that, to find a local estimate of the density SMCM takes into account the geometry of the data not just the count of number of points within a cut-off distance and hence it is likely to produce better building blocks. For example, consider a two dimensional data set having 31 points such that one point is at the center of a circle of radius 1 $\mathring{A}$ and the remaining thirty points are grouped into two clusters each having 15 points such that 10 points from each cluster are within a distance of 1 $\mathring{A}$ from the central point. Here TSCA will take the center point as a building block as it will have 21 points (including itself) within 1 $\mathring{A}$ while the remaining 5 points from each cluster will form two other clusters. Clearly these clusters and building blocks are not the desirable ones. But the SMCM will identify the center of each cluster with 15 points as the building block. These building blocks are better than those selected by TSCA because SMCM building blocks are at the centers of dense areas. The isolated central point will also be extracted as a building block but since it is supported by only one point, it is a poor building block and can be discarded. The SMCM is also expected to find better representative building blocks than hierarchical clustering or k-means type clustering. This is so because hierarchical clustering algorithms do not pay attention to the density of points (here density of similar structures). Moreover, a hierarchical clustering algorithm does not produce any prototypical building blocks. The poor performance of hierarchical clustering algorithms for fragment data is also pointed out in [14]. The usual k-means type clustering is also not very appropriate for such a problem as the mean of a set of 3-D structures (even after best alignment) will not have any associated residue sequence and hence is difficult to interpret.

4.4 Results

We have used the same 82 peptides as in [12]. (The list of peptides used can be found in Table 10.) To create the library of fragments, only the C_α coordinates are used. We use the same four proteins (1BP2, 1PCY, 4HHBb, 5PTI) as in [12] as the training set. The data in the Protein Data Bank (PDB) are updated as new information becomes available. As of December 2006, in PDB the information about the following 21 proteins was changed: 1APR (2APR), 1CPP (2CPP), 1CPV (5CPV), 1FB4h (2FB4h), 1FBJl (2FBJl), 1FDX (1DUR), 1GCR (4GCR), 1HMQa (2HMQa), 1INSa (4INSa), 1PCY (1PLC), 1SN3 (2SN3), 3PTP (5PTP), 3RXN (7RXN), 3TLN (8TLN), 4ADH (8ADH), 4ATCa (6AT1a), 1GAPa (1G6Na), 2FD1 (5FD1), 4FXN (2FOX), 2APP (3APP), 4CYTr (5CYTr). The new PDB ID is shown within parentheses. We have used both the old database as used in [12] and the new database downloaded in Dec. 2006.

4.4.1 Experimental Results

The SMCM has only one parameter, α. Using fragments of length 6 we have experimented with different choices of α such as α=2, 3, 4, 5 and 6 using the same database as used in [12]. We have found $\alpha = 4$ and 5 to yield better results. So we further fine-tuned α in the range 4 to 6 in steps of 0.5. Finally, we have got $\alpha = 5$ to produce the best result with a global-fit RMS 7.19 which is less than 7.3 that is reported in [12]. We have also experimented with the newly updated database. For the new data set $\alpha = 5.5$ resulted in the best global-fit RMS of 7.32. Table 17 summarizes the library size (number of building blocks) and the variations in the local-fit RMS error and global-fit RMS error with the choice of α for both data sets. To further compare the quality of building blocks, we have implemented TSCA method on the same data. We have obtained 55 main clusters and 102 sub-clusters (Unger et al. reported 103). SMCM extracted 104 building blocks. So for a fair comparison we remove the trailing 2 building blocks from 104 building blocks. Thus, for both methods we use the same number of building blocks to represent all target fragments and reconstruct the first 60 residues of 71 proteins whose lengths are larger than 60 residues using the same approach as in [12]. For our method, when we use only 102 clusters, the local-fit RMS increases to 0.75 and global-fit RMS increases to 7.23 which is still better than 7.3. Our implementation of TSCA results in a local-fit RMS of 0.77 and a global-fit RMS 8.27 and these are higher than the values reported in [12].

Table 17. Effect of the choice of α on the local-fit RMS error and the global-fit RMS error for the SMCM algorithm when the fragment length is six: (a) Original dataset and (b) Newly updated dataset

(a) Original Dataset A_{OLD}

α	Library Size	Local-fit RMS	Global-fit RMS
6	106	0.742	7.64
5.5	106	0.742	7.64
5	104	0.749	7.19
4.5	104	0.750	7.27
4	105	0.748	7.30
3.5	104	0.750	7.62
3	103	0.751	7.92
2	105	0.746	7.95

(b) Updated Dataset A_{NEW}

α	Library Size	Local-fit RMS	Global-fit RMS
8	108	0.726	7.53
7	107	0.727	7.48
6.5	107	0.727	7.48
6	107	0.723	7.32
5.5	107	0.723	7.32
5	107	0.725	7.59
4.5	107	0.727	7.69
4	106	0.728	7.75

4.4.2 Computation Complexity Analysis

Since the clustering algorithm is an iterative one, we provide an approximate analysis of its complexity. Note that, although the *Repeat –End Repeat* loop in the algorithm uses equation (2), $e^{-\alpha\, d^2(\mathbf{x}_k,\mathbf{x}_i)}$, $\forall i,k$ can be computed just once for all and stored in a table to be used in the repeat loop. Let us assume that the cost for computing $e^{-\alpha\, d^2(\mathbf{x}_k,\mathbf{x}_i)}$ for *any* pair $\{\mathbf{x}_k,\mathbf{x}_i\}$ be *D*. D involves the cost of computing the RMS deviation using the best molecular alignment of two segments, the cost of a multiplication, and that of computing the exponential. Thus, the total cost of computing $e^{-\alpha\, d^2(\mathbf{x}_k,\mathbf{x}_i)}, \forall i,k$ is $n(n-1)D/2$. The iterative part of the algorithm needs to make additions of such values read from the table. Let us assume that n hexamers are iteratively divided into c clusters and each cluster is of size k on average, i.e., $n = ck$. In the first iteration, the potential for each hexamer is calculated by summation of n exponential values taken from the table. It requires n^2 additions for n hexamers. In the second iteration, since k hexamers are removed and assigned to the first cluster, the remaining hexamers require $(n-k)^2$ additions in computation of potential. Likewise, in the final iteration, it requires k^2 additions. So the required time in addition operation to compute potential for the entire algorithm is:

$$\sum_{i=0}^{c-1}(n-ik)^2 A = k^2 \frac{c(c+1)(2c+1)}{6} A \qquad (7)$$

In (4) A is the cost of one table lookup and that of an addition operation. The total computation time required is thus, $T = \frac{n(n-1)}{2} D + k^2 \frac{c(c+1)(2c+1)}{6} A$. Since A is a constant and D is assumed to be a constant, the first term in T is of $O(n^2)$ and the second term is of $O(n^2c)$. For this specific dataset, SMCM takes 2 minutes 48 seconds on a personal computer with a 3.4GHz CPU and 1GB RAM to find the clusters while TSCA uses 51 seconds for the clustering task. One reason for the noticeable difference in running time between the two algorithms is that we did not use the efficient table lookup procedure but evaluated the distances and exponentials in every iteration of the repeat-loop in the algorithm. The use of small structural motifs as building blocks is based on the hypothesis that there are repeatedly occurring stable fragments and hence, in practice we shall not need to run such an algorithm on a very large database.

4.4.3 Visual Assessment of the Quality of the Building Blocks

To have a visual assessment of the quality of the building blocks and the reconstruction, in Figure 6(a) ~ Figure 6(c), we depict one of the most frequently used building block (NCYKQA), a *very good fit* target hexamer (LANWMC), and its representation using the building block. Figure 6(a) is the original building block (in solid red lines) while Figure 6(b) shows the original target hexamer (in dashed blue lines). Although at first sight the two structures look quite different, after the best alignment the superimposed structures look identical (Figure 6(c)). It is interesting to observe that the local secondary structures of both fragments are all alpha helix. This indicates that the identified building blocks are biologically meaningful structural motifs. Figure 6(d) displays another building block (NKEHKN) and Figure 6(e) depicts a *poor fit* target fragment (AAHCKN¸ the RMS error is larger than previous example but is still less than 1 Å). In this case too, we find a very good match between the two structures in Figure 6(f).

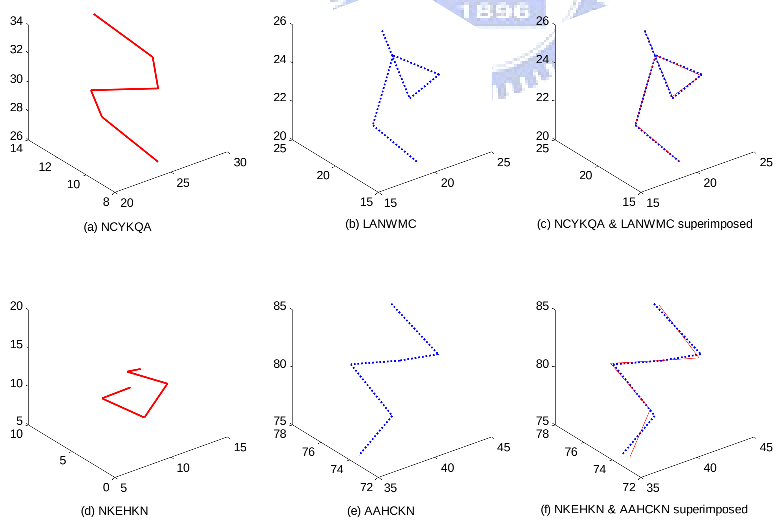


Figure 6. Representation of target fragments using building blocks.
(a) A building block with sequence NCYKQA; (b) A very good fit target hexamer with sequence LANWMC; (c) The building block and target hexamer superimposed after alignment; (d) Another building block with sequence NKEHKN; (e) A poor fit target hexamer with sequence AAHCKN; (f) The building block and target hexamer superimposed after alignment.

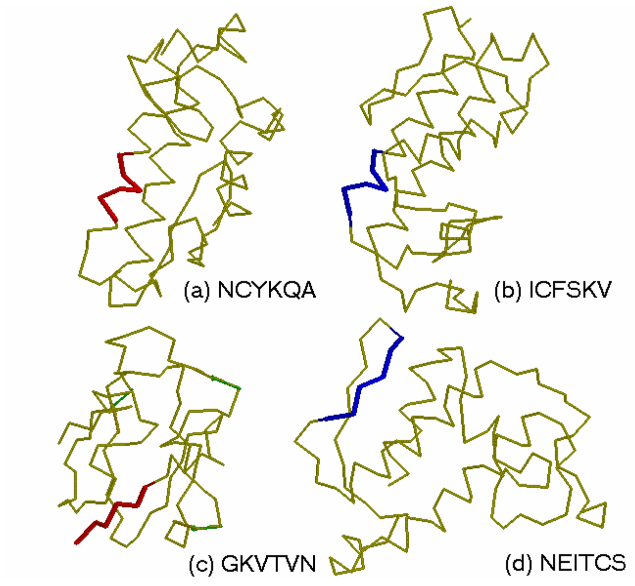


Figure 7. (a) SMCM building block NCYKQA at residue 50-55 of 1BP2; (b) TSCA building block ICFSKV at residue 104-109 of 1BP2; (c) SMCM building block GKVTVN at residue 94-99 of 1PCY; (d) TSCA building block NEITCS at residue 80-85 of 1BP2.

The most typical *helical* building block found by SMCM is NCYKQA and it is located at residue 50-55 of 1BP2; while the most populated building block found by TSCA [12] is ICFSKV and it is located at residue 104-109 of 1BP2. From the fact that ICFSKV is also an all helical structure and is included in the cluster of NCYKQA, it appears that the TSCA cluster associated with ICFSKV and the SMCM cluster associated with NCYKQA represent the same biological structural motif. Figure 7(a) and Figure 7(b) show these two building blocks and it is clear that they represent the same structural unit. Similarly, we find that the most typical extended strand GKVTVN found by SMCM is located at residue 94 of 1PCY while its counterpart NEITCS found by TSCA is located at residue 80 of 1BP2. These two building blocks are depicted in Figure 7(c) and Figure 7(d). The hexamer CSSENN is another interesting building block found by SMCM. Unger et al. [12] pointed out that CSSENN represents a turn joining two beta strands. Thus we find that building blocks found by SMCM represent structures of biological significance.

4.4.4 Alternative Ways of Performance Evaluation

To evaluate the local-fit RMS quality we compare the histograms of local-fit RMS error measuring the deviations of fragments from their corresponding building blocks (Figure 8(a)). It shows that the total count of *lower* RMS error (area under the curve) for SMCM is *larger* than that for TSCA and this indicates that more fragments are represented by good building blocks with lower reconstruction errors for SMCM.

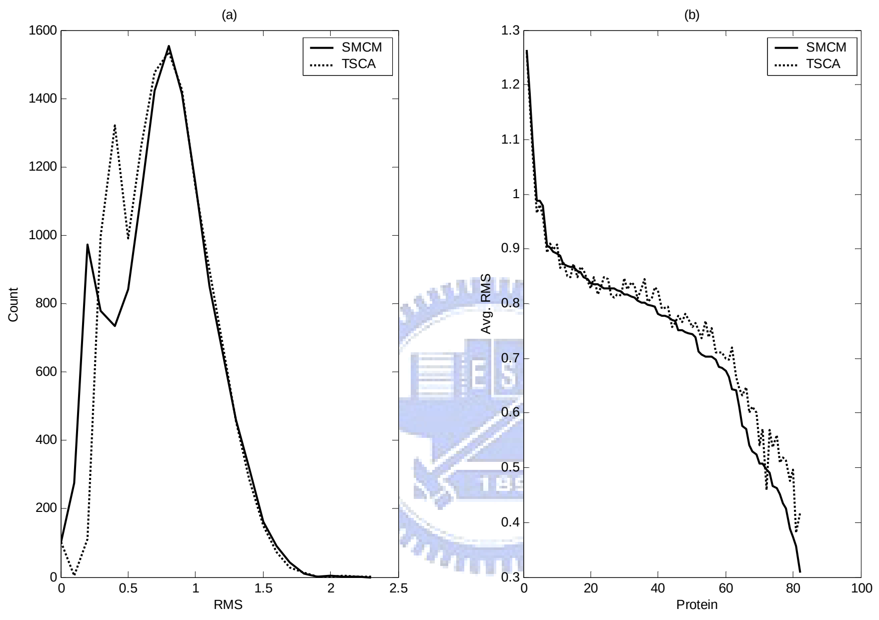


Figure 8. (a) Histograms of local-fit RMS errors for SMCM and TSCA (b) Protein by protein comparison of local-fit RMS error for SMCM and TSCA

Finally, we compare protein by protein the local-fit RMS error produced by SMCM and TSCA in Figure 8(b), where proteins are sorted in descending order of local-fit-RMS errors produced by SMCM. Figure 8(b) reveals that SMCM's local-fit RMS error is usually lower than the corresponding TSCA error. In this particular case, about 75% of the protein's SMCM local-fit RMS error is lower. The results on the updated database are found to be quite similar to the results on the original database. We have also experimented with fragment lengths 5, 6 and 7 and found that for the SMCM, the fragment length six is the optimal in terms of reconstruction error for both databases that we have experimented with. For SMCM the reconstruction error obtained with fragment length 7 is 7.57 which is slightly higher than that with hexamers. However, for TSCA the best reconstruction error of 7.59 is achieved with fragment length 7 while the error with fragment length 6 is 8.14.

4.4.5 Evaluation of the Library of Building Blocks on Other Datasets

To evaluate the quality of the library of building blocks, we use it to reconstruct proteins used in two more recent studies by Micheletti et al. [14] and Kolodny et al. [13]. We have excluded a few proteins with sequence discontinuity [51]. In these two datasets there are 10 and 144 proteins that are used for test, respectively. For the reconstruction, we follow a scheme similar in spirit with the method in [13]. The reconstruction process tries to minimize global-fit RMS deviation (RMSD). While, reconstructing residue by residue, instead of using the building block with the best local-fit RMS, the one with the minimum global-fit RMSD is chosen. It is noted that local-fit RMSD at each residue during such reconstruction usually will be higher. For the Micheletti et al. dataset, the global-fit RMSD obtained is 0.92 $\mathring{A}$ which is slightly lower than 1.06 $\mathring{A}$ reported in [14]. For the dataset used by Kolodny et al., authors reported the global-fit RMSD between 0.76 $\mathring{A}$ and 2.9 $\mathring{A}$ for different fragment lengths. While for this dataset, using hexamers we have achieved a global-fit RMSD of only 1.05 $\mathring{A}$ which is better than a global-fit RMSD of 1.26 $\mathring{A}$ reported in [13]. This further establishes that SMCM can extract biologically meaningful structural motifs that can be used for reconstruction of protein structure.

5. Incremental Structural Mountain Clustering Methods (ISMCM)

In case of a very big dataset for training, it becomes a time-consuming process for finding the clusters. To shorten the training time, an incremental approach is proposed. At first, we choose the longest protein in the training set and use it as the only protein for clustering to find the building blocks in the first step and evaluate the performance by checking the unassigned count of hexamers (that can't be assigned to any building block within 1 $\mathring{A}$) for each protein in the whole training set. The protein with the largest count of unassigned hexamers in this step is picked up and added to the selected set of proteins for clustering in the next step. Then, the two chosen proteins are used for clustering and select the next protein with high unassigned count. The same process is repeated until the unassigned ratio (abbreviated as U_ratio) of the whole set of hexamers is less than a threshold. Thus, we use only part of the original training dataset to cover the most occurring patterns and use them to find the building blocks accordingly. It will save computation time and complexity because the number of training fragments is reduced. Following the derivation in chapter 4, when the value n is reduced and then the approximate computation complexity of $O(n^2)$ and $O(n^2c)$ will also be reduced.

5.1 ISMCM Algorithm

Algorithm:

Input: Dataset P = {The complete list of proteins for training}

Choose: Threshold on unassigned ratio to stop the iteration

Initialization: Selected set: $P_1 = \{\}$, Remaining set: $P_2 = P$

```
Repeat until unassigned ratio is less than the threshold
    1. Move the protein with largest unassigned count from $P_2$ into $P_1$. Note that $P_1$ and $P_2$ satisfy the conditions: $P_1 \cup P_2 = P$ and $P_1 \cap P_2 = \{\}$ (for the first iteration, the longest protein is chosen and move into the selected set $P_1$)
    2. Find the building blocks from $P_1$ using SMCM.
    3. Compute the unassigned count of hexamers for each protein in $P_2$. These are the counts of hexamers that can't be represented by any building blocks derived from $P_1$ within a RMS error of 1 $\mathring{A}$. Also, compute the unassigned ratio of the whole set of hexamers.
End Repeat.
```

The incremental version of TSCA (ITSCA) can be written exactly in the same manner.

5.2. RESULTS

According to the incremental algorithm, we find the building blocks and evaluate the global-fit RMS (GRMS) and local-fit RMS (LRMS) for both Training set and Test set until U_ratio is less than some threshold. The results are given in the tables within this section.

5.2.1 Results on Dataset A

For the incremental version of the two algorithms, we have varied the fragment length from 5 to 7. The choice of α is also varied from 3.5 to 5.5. For each fragment length, we report results with the best choices of α. Table 18 summarizes the results using the ISMCM algorithm for Dataset A_{OLD}. In Table 18, U_ratio denotes the percentage of total fragments that cannot be assigned to any building block within a distance of 1 $\mathring{A}$. In this case too, we find that fragment length 6 with α=5 yields the best result of global-fit RMS error 7.19 which is less than 7.3 reported in Unger et al. These results are produced using the same dataset as used in Unger et al. [12].

Table 18. ISMCM results on Dataset A_{OLD}

Frag. length	α	Protein count	PDB No.	Library size	U_ratio	Train LRMS	Train GRMS	Test LRMS	Test GRMS
5	3.5	1	4HHBb	20	20.5%	0.60	7.86	0.81	10.53
5	3.5	2	1PCY	38	2.9%	0.44	6.64	0.62	8.67
5	3.5	3	1BP2	40	1.5%	0.43	7.60	0.61	8.07
5	3.5	4	5PTI	44	0.0%	0.42	5.90	0.60	8.08
6	5.0	1	4HHBb	35	35.0%	0.77	5.80	1.08	9.48
6	5.0	2	1PCY	73	9.4%	0.49	7.06	0.81	8.79
6	5.0	3	1BP2	93	3.0%	0.41	5.07	0.76	8.36
6	5.0	4	5PTI	104	0.0%	0.37	3.35	0.75	7.19
7	3.5	1	4HHBb	51	43.8%	0.95	8.68	1.38	10.16
7	3.5	2	1PCY	117	16.4%	0.49	4.67	0.98	8.27
7	3.5	3	1BP2	153	6.2%	0.36	3.69	0.93	7.95
7	3.5	4	5PTI	173	0.0%	0.28	2.19	0.90	7.60

Table 18 shows that with one protein in the training set, the test error is quite high. As we increase the number of proteins in the training set, the number of building blocks increases and the training and test errors decrease. Table 18 also reveals that increasing the number of proteins from 1 to 2 in training set changes the number of building blocks and test error more drastically than those by increasing the number of training proteins from 3 to 4. This asymptotic behavior, which will be illustrated further with Dataset B, indicates the utility and consistency of the incremental version of SMCM.

Table 19 depicts the performance of the ISMCM on the updated version of Dataset $A_{NEW.}$ We find from Table 19 that when sequence length=6 and α=5.5, we have the best reconstruction errors on the test set using 107 clusters: local-fit RMS = 0.72 and global-fit RMS = 7.32. On the other hand, when we apply incremental version of the TSCA to the same dataset with sequence length 6 and use the 4 training proteins to construct the building blocks, we obtain 101 clusters with no unassigned hexamers (unassigned ratio =0%). The local-fit RMS error is 0.76 and global-fit RMS error is 8.14 on the test data (See Table 20). Since ISMCM uses six more building blocks than ITSCA, to make a fair comparison of ITSCA and ISMCM, we remove the trailing 6 building blocks from the 107 building blocks. Thus, for both methods we now use the same number of building blocks to represent all target fragments and reconstruct the first 60 residues of the 71 proteins whose lengths are larger than 60. For ISMCM, when we use only 101 clusters, the local-fit RMS very marginally increases to 0.73 and global-fit RMS increases to 7.55 from 7.32 (a 3% increase). But it is still better than 8.14 realized by ITSCA with the same fragment length of six.

Table 19. ISMCM results on the updated Dataset A_{NEW}

Frag. length	α	Protein count	PDB No.	Library size	U_ratio	Train LRMS	Train GRMS	Test LRMS	Test GRMS
5	5.5	1	4HHBb	20	20.7%	0.60	7.88	0.79	10.37
5	5.5	2	1PCY	35	3.9%	0.44	7.76	0.61	9.14
5	5.5	3	1BP2	39	0.7%	0.42	5.14	0.59	7.97
5	5.5	4	5PTI	45	0.0%	0.41	4.63	0.59	8.86
6	5.5	1	4HHBb	35	35.0%	0.77	6.60	1.06	9.63
6	5.5	2	1PCY	74	9.9%	0.49	6.59	0.79	8.29
6	5.5	3	1BP2	94	3.2%	0.41	4.92	0.74	8.21
6	5.5	4	5PTI	107	0.0%	0.36	4.00	0.72	7.32
7	3.5	1	4HHBb	51	43.8%	0.95	8.28	1.36	10.06
7	3.5	2	1PCY	117	16.7%	0.49	4.46	0.96	8.03
7	3.5	3	1BP2	152	6.2%	0.36	2.76	0.91	7.83
7	3.5	4	5PTI	174	0.0%	0.28	1.90	0.88	7.57

When we apply the ITSCA to the updated Dataset A_{NEW}, we get the best result with fragment length=7 and using all 4 proteins. However, the test global-fit RMS error is 7.59, which is still higher than the global-fit RMS error of 7.32 produced by the SMCM with fragment length 6. The results are summarized in Table 20.

Table 20. ITSCA results on the updated Dataset A_{NEW}

Frag. Length	Protein count	PDB No.	Library size	U_ratio	Train LRMS	Train GRMS	Test LRMS	Test GRMS
5	1	4HHBb	18	21.0%	0.69	8.28	0.86	10.81
5	2	1PCY	29	4.6%	0.57	7.48	0.70	9.07
5	3	1BP2	34	2.2%	0.50	8.00	0.64	9.04
5	4	5PTI	38	0.0%	0.49	5.64	0.62	8.21
6	1	4HHBb	34	34.7%	0.81	7.39	1.08	10.12
6	2	1PCY	70	9.9%	0.53	7.33	0.82	8.83
6	3	1BP2	90	3.4%	0.46	6.63	0.77	8.29
6	4	5PTI	101	0.0%	0.43	5.94	0.76	8.14
7	1	4HHBb	51	43.3%	0.96	8.62	1.37	10.05
7	2	1PCY	113	16.4%	0.51	4.67	0.98	8.31
7	3	1BP2	151	6.2%	0.38	3.47	0.92	7.95
7	4	5PTI	168	0.0%	0.31	2.12	0.90	7.59

5.2.2. Results on the Dataset B

For this data too, we have experimented with fragment lengths 5, 6 and 7 as summarized in Table 21 and Table 22 for the ISMCM and ITSCA respectively. For these two tables we find that ISMCM with fragment length 7 produces the best results of global-fit RMS error of 14.67 which is better than the best global-fit RMS error of 16.26 achieved by ITSCA with fragment length seven. However, the ISMCM usually finds more building blocks than the ITSCA. For example, Table 22 shows that with fragment length 7 and five training proteins, the total number of building blocks found by ITSCA is 756 and this results in a global-fit RMS reconstruction error of 16.26 whereas for ISMCM the number of building blocks is 871 yielding the best reconstruction error of 14.67. Just to compare the performance when ISMCM uses 716 building locks (fragment length 7, number of proteins equal to 4) the global-fit RMS error is 15.44 which is again smaller than 16.26. Thus the improvement in performance by the

ISMCM is primarily not by the fact that it finds and uses more building blocks but because of quality of the building blocks that are placed at the center of dense areas of data points (here 3-D structures of length 5, 6 or 7).

Table 21. ISMCM results on Dataset B

Frag. length	α	Protein count	PDB No.	Library size	U_ratio	Train LRMS	Train GRMS	Test LRMS	Test GRMS
5	5.5	1	1YGE	61	2.7%	0.52	17.02	0.58	19.78
5	5.5	2	1CZFa	73	1.7%	0.52	17.36	0.58	19.49
5	5.5	3	1DMR	81	1.0%	0.52	17.23	0.57	19.38
5	5.5	4	1SMD	86	0.7%	0.50	16.71	0.55	18.63
5	5.5	5	1LAM	91	0.5%	0.50	16.75	0.55	17.83
5	5.5	6	1PPN	92	0.4%	0.50	16.43	0.55	17.93
6	5	1	1YGE	171	11.0%	0.62	15.93	0.69	18.03
6	5	2	1DMR	221	6.5%	0.59	15.44	0.66	17.20
6	5	3	1CZFa	258	5.0%	0.61	15.58	0.67	17.26
6	5	4	1SMD	288	4.1%	0.59	14.38	0.65	16.70
6	5	5	1B4Va	316	3.4%	0.58	14.46	0.64	16.06
6	5	6	3SIL	354	2.8%	0.57	14.44	0.64	16.30
7	5	1	1YGE	337	27.8%	0.76	15.51	0.83	17.52
7	5	2	1DMR	517	19.5%	0.70	14.29	0.78	15.95
7	5	3	1CZFa	614	16.7%	0.68	14.13	0.76	16.59
7	5	4	1SMD	716	14.3%	0.67	13.76	0.75	15.44
7	5	5	1KAPp	798	12.3%	0.65	13.39	0.74	14.91
7	5	6	3SIL	871	11.0%	0.64	13.39	0.73	14.67

Comparison of the global-fit RMS errors in Tables 21 and 22 reveals that the ISMCM errors are usually less than those by the ITSCA. From Table 21 we also find that as we increase the number of training proteins, the number of building blocks increases. But the increase in the number of building blocks when the number of training proteins is increased from 1 to 2 is much more than that when we increase the number from 4 to 5. And this is true for all fragment lengths. Moreover, going beyond five proteins increases the number of building blocks only marginally. These are very desirable attributes of any incremental algorithm and it suggests that beyond certain number, increasing the number of proteins in the training data will not have much effect on the building blocks.

Table 22. ITSCA results on Dataset B

Frag. length	Protein count	PDB No.	Library size	U_ratio	Train LRMS	Train GRMS	Test LRMS	Test GRMS
5	1	1YGE	56	3.1%	0.62	19.29	0.66	20.83
5	2	1DMR	56	2.0%	0.61	18.38	0.64	20.31
5	3	1CZFa	70	1.3%	0.56	17.36	0.60	19.23
5	4	1LAM	74	0.9%	0.60	18.88	0.63	20.27
5	5	1SMD	80	0.8%	0.60	18.77	0.63	20.23
5	6	1BXOa	79	0.6%	0.59	18.50	0.63	20.01
6	1	1YGE	151	13.7%	0.68	16.68	0.74	18.90
6	2	1DMR	208	7.9%	0.68	15.44	0.73	17.60
6	3	1CZFa	239	6.2%	0.67	15.27	0.72	17.14
6	4	1SMD	274	4.9%	0.66	15.39	0.71	17.25
6	5	1LAM	289	4.4%	0.61	15.15	0.67	16.47
6	6	1QKSa	306	3.7%	0.60	15.22	0.66	16.66
7	1	1YGE	327	29.7%	0.80	16.18	0.86	17.76
7	2	1DMR	496	20.4%	0.79	16.09	0.84	17.26
7	3	1CZFa	577	17.9%	0.77	15.49	0.83	16.65
7	4	1SMD	685	15.0%	0.75	15.66	0.82	16.91
7	5	1VNS	756	13.3%	0.74	15.22	0.81	16.26
7	6	1QKSa	806	12.2%	0.73	15.17	0.80	16.30

To investigate it further, in Table 23 we report the results when we increased the number of proteins in the training set to 12 with fragment length 6. Table 23 depicts that the first six proteins generated 354 building blocks whereas another additional six proteins added only 68 building blocks. When the number of proteins is increased from 11 to 12 the number of building blocks is increased by just 1. This behavior is more clearly reflected in Figure 9 which displays the variation of number of building blocks and global-fit RMS errors on the test data as a function of number of proteins in the training set. Thus use of more proteins in the training data increases the computational cost substantially, but the gain may not be significant. The computational cost will be increasing when more proteins are used for training and the marginal benefit is decreasing. As we can find in the Figure 9, the library size is going to stop increasing its number, and also the test GRMS stop decreasing. The details are listed in the Table 23.

Table 23. ISMCM results on Dataset B using 12 training proteins

Frag. Length	α	Protein count	PDB No.	Library Size	U_ratio	Train LRMS	Train GRMS	Test LRMS	Test GRMS
6	5	1	1YGE	171	11.0%	0.62	15.93	0.69	18.03
6	5	2	1DMR	221	6.5%	0.59	15.44	0.66	17.20
6	5	3	1CZFa	258	5.0%	0.61	15.58	0.67	17.26
6	5	4	1SMD	288	4.1%	0.59	14.38	0.65	16.70
6	5	5	1B4Va	316	3.4%	0.58	14.46	0.64	16.06
6	5	6	3SIL	354	2.8%	0.57	14.44	0.64	16.30
6	5	7	1LAM	372	2.6%	0.56	14.14	0.63	16.60
6	5	8	1QGUa	387	2.2%	0.56	14.57	0.63	16.12
6	5	9	1DGFa	394	2.1%	0.56	14.65	0.63	16.35
6	5	10	7A3Ha	411	1.7%	0.55	14.00	0.62	16.08
6	5	11	1EZM	421	1.7%	0.55	14.57	0.62	15.76
6	5	12	1KAPp	422	1.7%	0.56	14.72	0.62	15.91

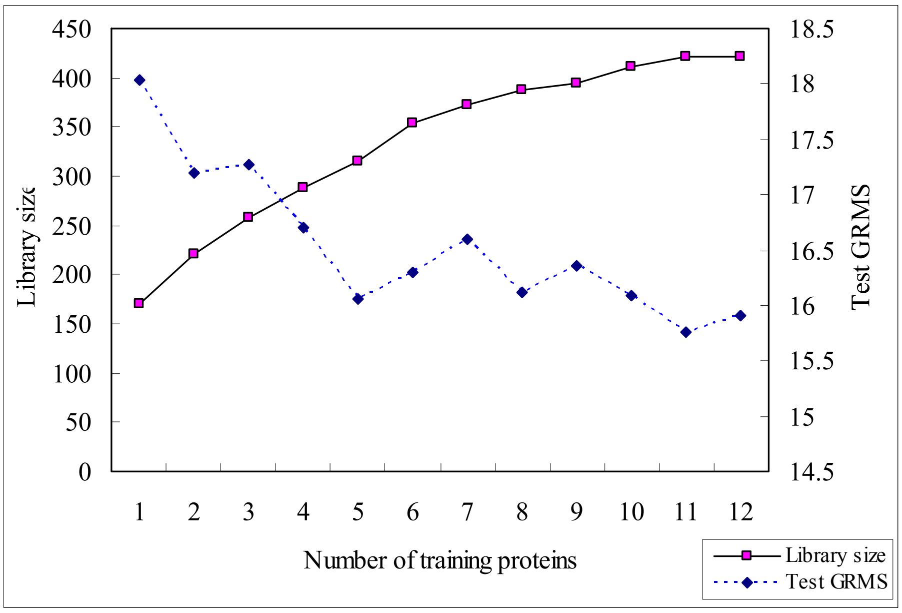


Figure 9. The variation of library size and that of reconstruction errors as functions of the number of training proteins.

5.2.3 Alternative Ways of Performance Evaluation

To evaluate quality of the building blocks for Dataset A_{NEW}, we compare the histogram of local-fit RMS measuring the deviations of fragments from their corresponding building blocks (Figure 10(a)). It is found that the total count of *lower* RMS error (area under the curve) for ISMCM is *larger* than that for ITSCA and this indicates that more fragments are represented by good building blocks with lower errors for ISMCM. Finally, we sort the proteins according to their ISMCM local-fit RMS deviations and compare the results one by one with the ITSCA local-fit RMS error for the same protein. We get the two curves as shown in Figure 10(b), which reveals that ISMCM local-fit RMS errors are usually lower than ITSCA errors.

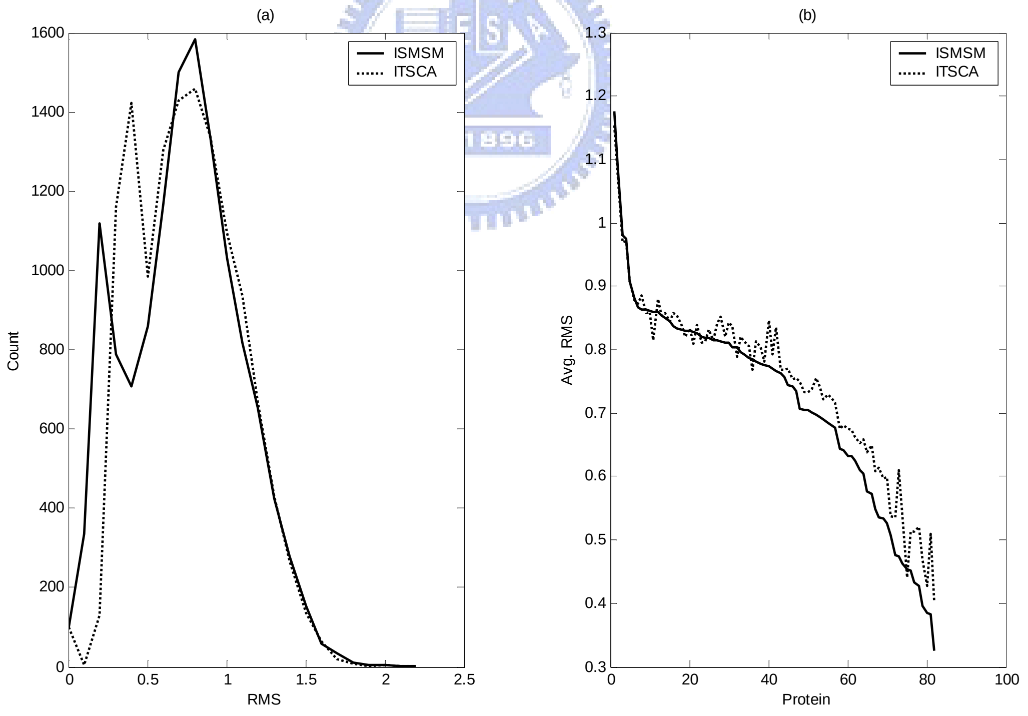


Figure 10. (a) Histograms of local-fit RMS errors for ISMCM and ITSCA (b) Protein by protein comparison of local-fit RMS error for ISMCM and ITSCA on Dataset A_{NEW}

To compare the performance of both methods on the Dataset B, we proceed in the same way as we did for Dataset A. We remove the trailing clusters with smaller number of members to make both methods use the same number of building blocks. In Figure 11(a) and 11(b), we compare the histogram of local-fit RMS errors and average local-fit RMS error per protein, respectively. Like Dataset A, here we also find that ISMCM outperforms ITSCA with respect to these evaluation criteria.

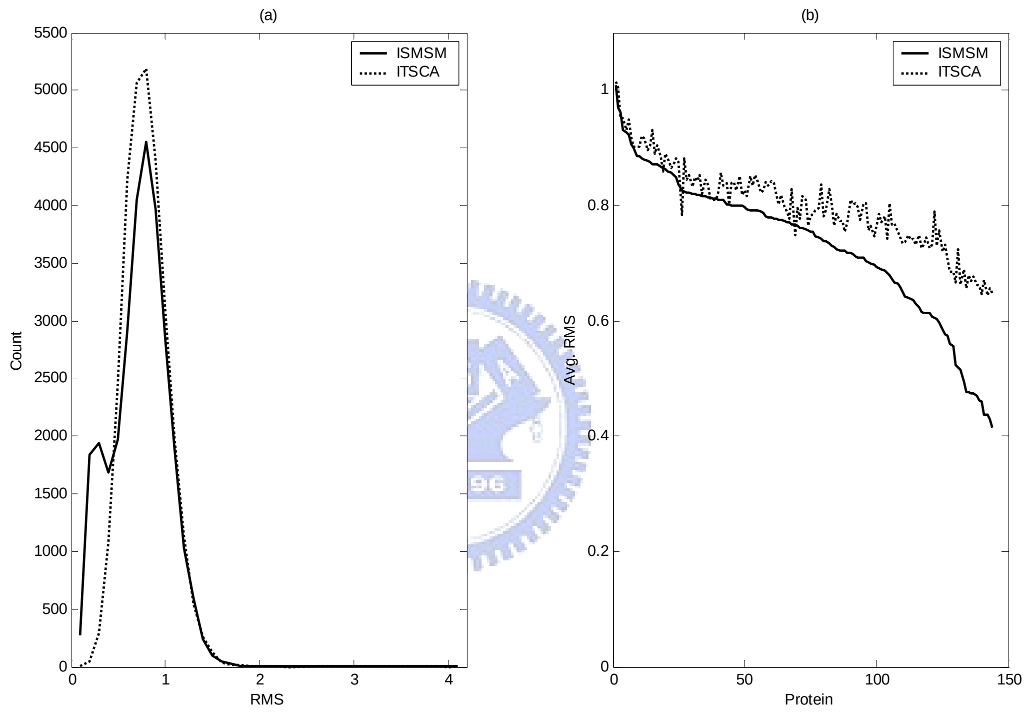


Fig.ure 11. (a) Histograms of local-fit RMS errors for ISMCM and ITSCA (b) Protein by protein comparison of local-fit RMS error for ISMCM and ITSCA on Dataset B

5.2.4 Visual Assessment of the Quality of the Building Blocks

In this section, we examine visually how well the building blocks can represent the target fragments. For this we consider two examples, one with a very good fit (Figure 12) and the other with a relatively poor fit (Figure 13), but still within 1 $\overset{\circ}{A}$ threshold. Figure 12 (a) shows a building block (AVGFMLA) whereas Figure 12(b) represents a target fragment (TLSELHC).

Apparently the two structures look quite different. But Figure 12(c), the rotated version of the building block AVGFMLA obtained after best molecular fit with the target look almost identical to Figure 12(b). The superimposition of AVGFMLA and TLSELHC shown in Figure 12(d) clearly demonstrates an excellent fit between the two. The four panels in Figure 13 show the representation of the target fragment EGVEIAC with the building block ENAIGGS. Although, in terms of the local-fit RMS error it is a poorer fit, yet the building block matches very nicely to the target (Figure 13(d)).

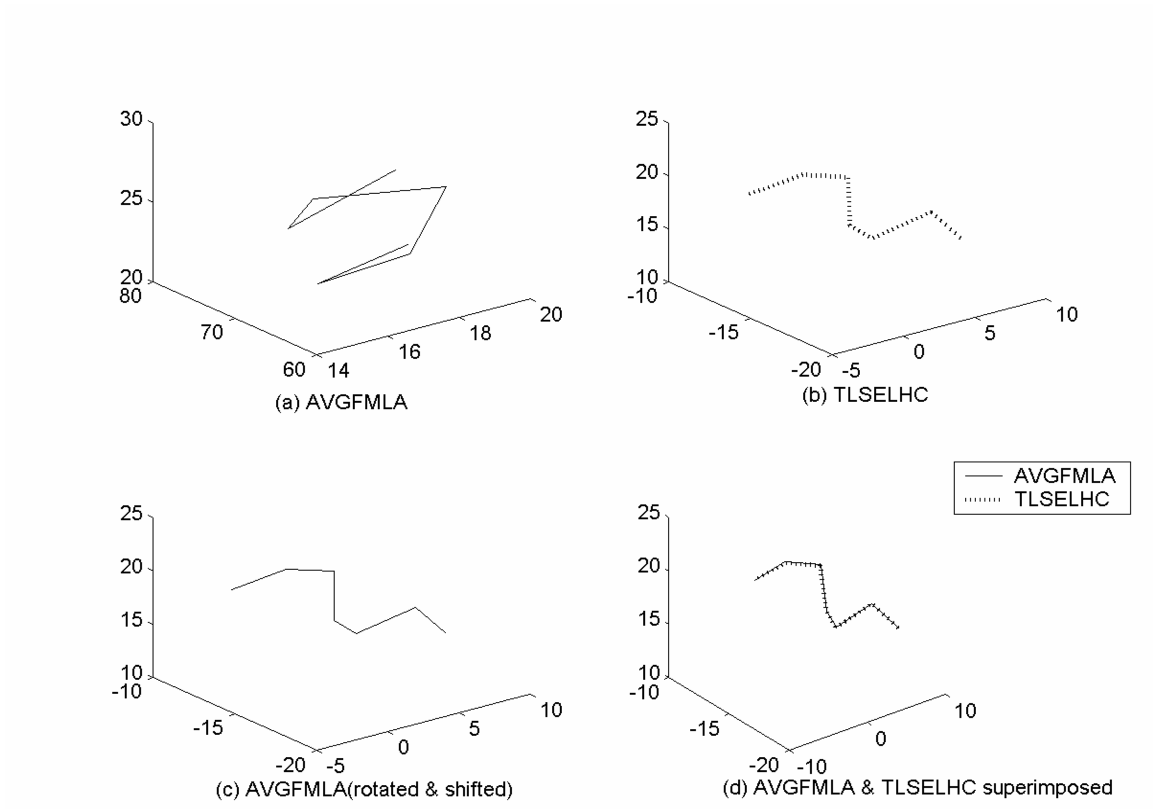


Figure 12. Representation of a target fragment using a building block, a good fit case. (a) The original building block; (b) The target hexamer; (c) The rotated and shifted building block; (d) The building block and target hexamer superimposed after alignment.

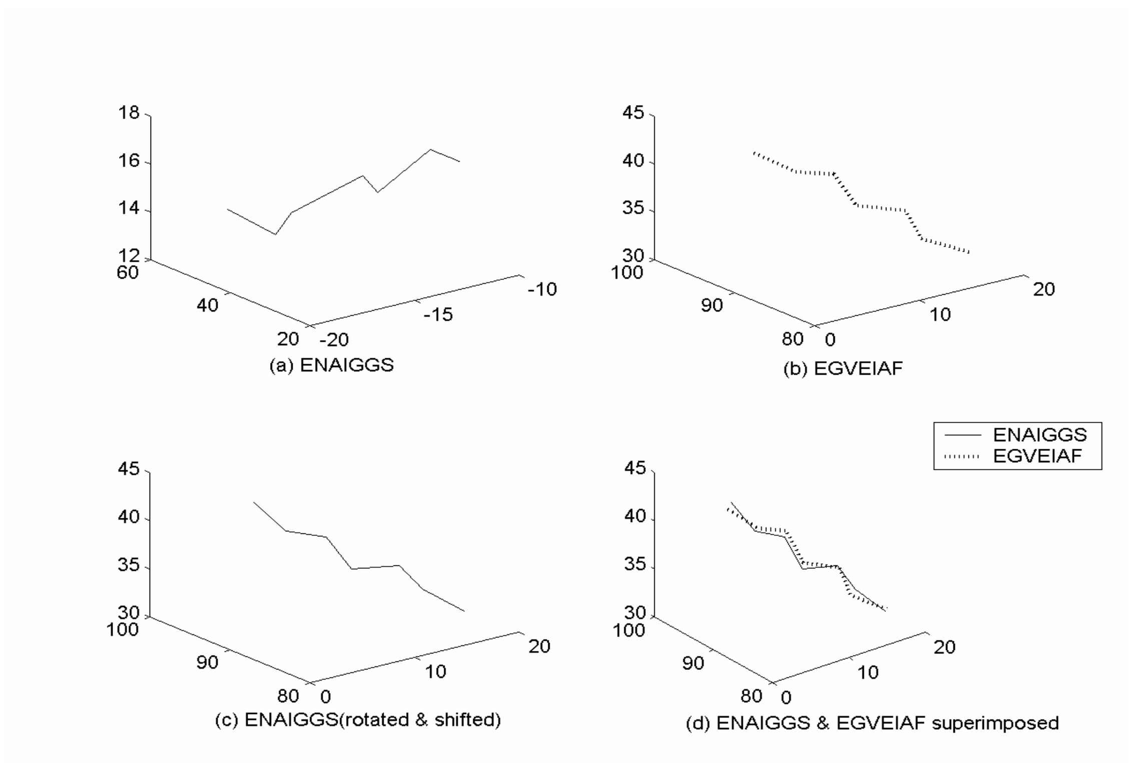


Figure 13. Representation of a target fragment using a building block, a poor fit case. (a) The original building block; (b) The target hexamer; (c) The rotated and shifted building block; (d) The building block and target hexamer superimposed after alignment.

Next, we would like to show the biological structures of the building blocks of top two most populated clusters using ISMCM and compare them with the building blocks of top two clusters using ITSCA method. The most typical helical building block found by ISMCM is AVGFMLA and it is located at residue 324-330 of 1SMD; whereas the most populated building block found by ITSCA is GAAQVIM and it is located at residue 147-153 of 1DMR. The fact that GAAQVIM is also helical structure and is included in the cluster of AVGFMLA, appears that the ITSCA cluster associated with GAAQVIM and the ISMCM cluster associated with AVGFMLA represent the same biological structural motif. Figure 14(a) and Figure 14(b) show these two building blocks and it is clear that they represent the same structural unit. Similarly, we find that the most typical extended strand TKVIFEG found by ISMCM is located at residue 43 of 1CZFa whereas its counterpart GIKIYVS found by ITSCA is located at residue 464 of 1SMD. These two building blocks are depicted in Figure 15(a) and Figure 15(b). It can be seen that these building blocks represent similar structures of biological significance.

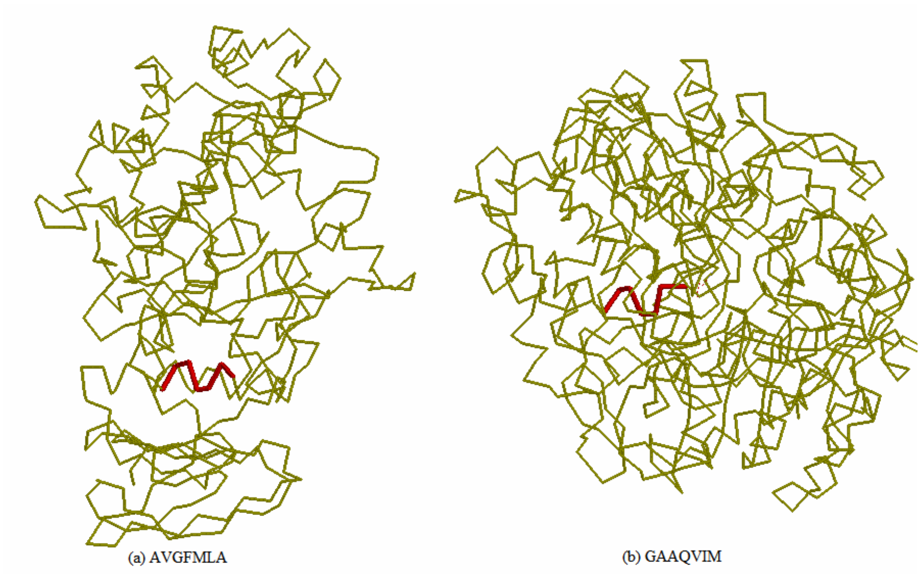


Figure 14. (a) ISMCM building block (AVGFMLA) at residue 324-330 of 1SMD (b) ITSCA building block (GAAQVIM) at residue 147-153 of 1DMR

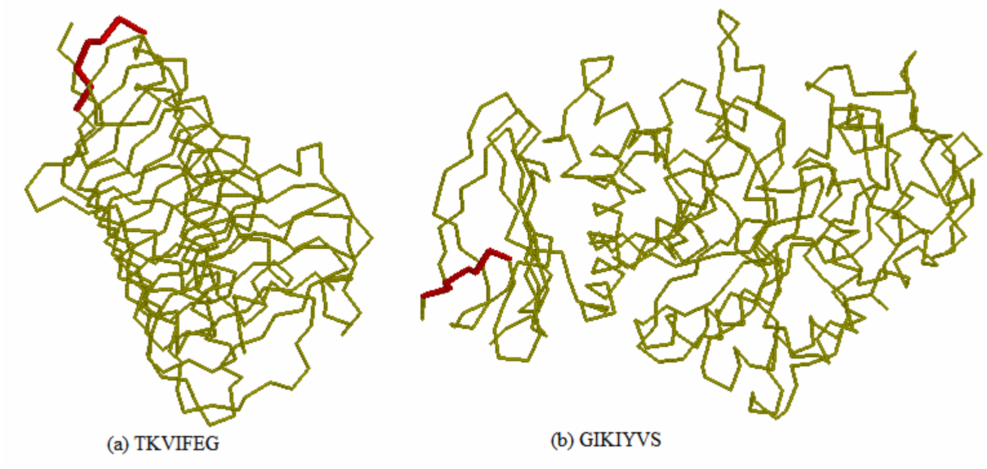


Figure 15. (a) ISMCM building block (TKVIFEG) at residue 43-49 of 1CZFa (b) ITSCA building block (GIKIYVS) at residue 464-470 of 1SMD

6. CONCLUSIONS AND FUTURE WORK

We have applied the concept of combinatorial fusion to improve accuracy in protein structure prediction. In particular, we have successfully improved the overall predictive accuracy rate of 87% for the four classes and 69.6% for the 27 folding patterns. We improve previous results by Huang et al. [9] (65.5% for folding structures) and Ding and Dubchak [8] (56.5% for folding structures) by incorporating the method of combinatorial fusion with the RBFN neural network using the hierarchical learning architecture. These rates are higher than previous results and it demonstrates that data fusion is a viable method for feature selection and combination in the prediction and classification of protein structures. Work has been performed to improve those results which used other machine learning technique such as kernel method, SVM and genetic algorithm. For example, Yu et al. [43] has obtained good accuracy rate using SVM with *n*-peptide coding schemes and jury voting. Future work can be performed to improve these results using our combinatorial fusion approach.

Also, we present a structural variant of the mountain clustering method that is suitable for data like 3-D structures of protein fragments. We have analyzed the SMCM and TSCA and have demonstrated that since TSCA does not take into account the geometry of the data, it may extract poorer building blocks than the SMCM. The utility of this algorithm is demonstrated on the same dataset used by Unger et al. In fact, the superiority of this algorithm is demonstrated on two versions of datasets (the original one and the newly updated one on the same set of proteins). To visually compare the quality of reconstructions we also proposed two alternative ways revealing that the performance of SMCM building blocks is usually better than TSCA building blocks both in terms of the local-fit RMS histogram and in terms of the average RMS deviation for individual protein. Our experiments demonstrate that the SMCM can find useful building blocks to successfully reconstruct the 3-D protein structures for the first 60 residues (as done by Unger et al.) of all test proteins with global-fit RMS error within 7.19 $\mathring{A}$. It can also obtain good local-fit RMS errors indicating that these building blocks can model the nearby fragments within tolerable errors.

Both SMCM and TSCA are computationally expensive when the size of training dataset is large. Hence we proposed an incremental version of the SMCM. The same concept is also used to obtain an incremental version of the TSCA. We have made extensive experimentation with these two algorithms using two versions of the dataset used by Unger et al. as well as another dataset used by other researchers. The incremental SMCM is also found to be quite effective and it is found to exhibit the properties expected from an incremental algorithm. More specifically, as the number of proteins increases in the training set, the increase in the number of building blocks decreases and consequently the rate of decrease in the global reconstruction error both on the training and test data falls down. Moreover, the incremental SMCM is found to be more effective than the incremental TSCA. Although, the SMCM usually finds more building blocks than those found by the TSCA, we have demonstrated that the improved performance for SMCM comes from the quality of the building blocks which are placed at the center of areas dense in training data.

None of the algorithms discussed here can take into account fragments of variable length. To extend the algorithms for fragments of variable length, we need measures of similarity between fragments of different lengths. For example, if we have two fragments both are helix, but of different length, the structural similarity between the two should be very high; on a [0-1] scale, it should be 1. We plan to investigate this in near future.

Bibliography

[1] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia, "SCOP: a structural classification of proteins database for the investigation of sequence and structures," Journal of Molecular Biology, Vol. 247, pp. 536-540, 1995.

[2] C. A. Orengo, A. D. Michie, S. Jones, D. T. Jones, M. B. Swindells, and J. M. Thornton, "CATH – a hierarchic classification of protein domain structure," Structure, Vol. 5, No. 8, pp. 1093-1108, 1997.

[3] R.D. Finn, J. Tate, J. Mistry, P.C. Coggill, J.S. Sammut, H.R. Hotz, G. Ceric, K. Forslund, S.R. Eddy, E.L. Sonnhammer and A. Bateman," The Pfam protein families database," Nucleic Acids Research: Database Issue, Vol. 36, pp. D281-D288, 2008.

[4] D. Baker and A. Sali, "Protein structure prediction and structural genomics," Science, Vol. 294, pp.93-96, 2001.

[5] I. Dubchak, I. Muchnik, S. R. Holbrook, and S. H. Kim, "Prediction of protein folding class using global description of amino acid sequence," Proc. Natl. Acad. Sci., USA, Vol. 92, pp. 8700-8704, 1995.

[6] K.C. Chou and C.T. Zhang, "Prediction of protein structural classes," Crit. Rev. in Biochem. Mol. Biol., Vol. 30, No. 4, 1995, pp. 275-349.

[7] A. Antonina, H. Dave, E.B. Steven, J.P.H. Tim, C. Cyrus, and G.M. Alexey, "SCOP database in 2004: refinements integrate structure and sequence family data," Nuclear Acid Research, Vol. 32, 2004, pp.226-229.

[8] C.H.Q. Ding and I. Dubchak, "Multi-class protein fold recognition using support vector machines and neural networks," Bioinformatics, Vol. 17, No. 4, 2001, pp. 349-358.

[9] C. D. Huang, C.T. Lin, and N.R. Pal, "Hierarchical learning architecture with automatic feature selection for multi-class protein fold classification," *IEEE Trans. NanoBioscience*, Vol. 2, No. 4, 2003, pp. 503-517.

[10] J.M. Bujnicki, "Protein structure prediction by recombination of fragments," *ChemBioChem*, Vol. 7, pp. 19-27, 2006.

[11] N. Haspel, C. J. Tsai, H. Wolfson and R. Nussinov, "Hierarchical protein folding pathways: A computational study of protein fragments," *Proteins: Structure, Function, and Genetics*, Vol.51, Issue 2, pp. 203-215, 2003.

[12] R. Unger, D. Harel, S. Wherland and J.L. Sussman, "A 3D building blocks approach to analyzing and predicting structure of proteins," *Proteins: Structure, Function, and Genetics*, Vol. 5, pp. 355–373, 1989.

[13] C. Micheletti, F. Seno and A. Maritan, "Recurrent oligomers in proteins: an optimal scheme reconciling accurate and concise backbone representations in automated folding and design studies," *Proteins: Structure, Function, and Genetics*, Vol. 40, pp. 662–674, 2000.

[14] R. Kolodny, P. Koehl, L. Guibas and M. Levitt, "Small libraries of protein fragments model native protein structures accurately," *Journal of Molecular Biology*, Vol. 323, pp. 297–307, 2002.

[15] R.R. Yager and D.P. Filev, "Approximate clustering via the mountain method," *IEEE Trans. Systems Man and Cybernetics*, Vol. 24, pp. 1279-1284, 1994.

[16] S.L. Chiu, "Extracting fuzzy rules for pattern classification by cluster estimation," in Proc. 6th Int. Fuz. Systs. Assoc, World Congress (IFSA'95), 1995, pp. 1-4.

[17] K. L. Lin, C. T. Lin, N. R. Pal, and S. Ojha, "Finding useful building blocks for

construction of protein 3-D structures using a structural variant of mountain clustering method," in Second IAPR International Workshop (PRIB 2007), submitted for publication.

[18] L. L. Conte, B. Ailey, T. J. Hubbard, S. E. Brenner, A. G. Murzin, and C. Chothia, "SCOP: a structural classification of proteins database," *Nuclear Acid Research*, Vol. 28, No. 1, pp. 257-259, 2000.

[19] L. L. Conte, S. E. Brenner, T. J. Hubbard, C. Chothia, and A.G. Murzin, "SCOP database in 2002: refinements accommodate structural genomics," *Nucleic Acids Research*, Vol. 30, No. 1, pp. 264-267, 2002.

[20] A. Andreeva, D. Howorth, J.M. Chandonia, S.E. Brenner, T.J.P. Hubbard, C.Chothis and A. G. Murzin, "Data growth and its impact on the SCOP database: new developments", *Nucleic Acids Research*,Vol. 36, D419-D425, 2008

[21] I. Dubchak, I. Muchnik, C. Mayor, I. Dralyuk, and S. H. Kim "Recognition of a protein fold in the context of the SCOP classification," *Proteins: Structure, Function, and Genetics*, Vol. 35, pp. 401-407, 1999.

[22] F.M. Pearl, D. Lee, J.E. Bray, I. Sillitoe, A.E. Todd, A.P. Harrison, J.M. Thornton, and C.A. Orengo, "Assigning genomic sequences to CATH," *Nuclear Acid Research*, Vol. 28, No. 2, 2000, pp. 584-599.

[23] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, N.Y., 1995.

[24] D.F. Hsu, J. Shapiro, and I. Taksa, "Methods of data fusion in information retreival: rank vs. score combination," *DIMACS Technical Report 58*, 2002.

[25] D.F. Hsu and I. Taksa, "Comparing rank and score combination methods for data fusion in information retrieval," *Information Retrieval*, Vol. 8, 2005, pp. 449-480.

[26] J.M. Yang, Y.F. Chen, T.W. Shen, B.S. Kristal, and D.F. Hsu, “Consensus scoring criteria for improving enrichment in virtual screening,” Journal of Chemical Information and Modeling, Vol. 45, 2005, pp. 1134-1146.

[27] D.F. Hsu, Y.S. Chung, and B.S. Kristal, “Combinatorial fusion analysis: method and practice of combining multiple scoring systems,” Advanced Data Mining Technologies in Bioinformatics, Idea Group Inc., 2006, pp. 32-36.

[28] K.B. Ng and P.B. Kantor, “Predicting the effectiveness of naïve data fusion on the basis of system characteristics,” J. American Society for Information Sci., Vol. 51. No. 13, 2000, pp. 1177:1189.

[29] P. Baldi and S. Brunak, Bioinformatics: the Machine Learning Approach, MIT Press, 1998.

[30] C.H. Wu, Neural Networks and Genome Informatics. Amsterdam, The Netherlands: Elsevier, 2000.

[31] J. Moody and C. J. Darken, “Fast learning in networks of locally tuned processing units,” Neural Computation, Vol. 1, No. 2, pp. 281-294, 1989.

[32] N.J. Belken, P.B. Kantor, E.A. Fox, and J.A. Shaw, “Combining evidence of multiple query representation for information retrieval,” Information Processing and Management, Vol. 31, No. 3, 1995, pp. 431-448.

[33] C.C. Vogt and G.W. Cotrell, “Fusion via a linear combination of scores,” Information Retrieval, Vol. 1, 1999, pp. 151-172.

[34] L. Xu, A. Krzyzak, and C.Y. Suen, “Method of combining multiple classifiers and their application to handwriting recognition,” IEEE Trans. Systems Man and Cybernetics, Vol. 22,

1992, pp. 418-435.

[35] C.M.R. Ginn, P. Willett, and J. Bradshaw, "Combination of molecular similarity measures using data fusion," Perspectives in Drug Discovery and Design, Vol. 20, 2000, pp.1-16.

[36] M.A. Kuriakose, W.T. Chen, Z.M. He, A.G. Sikora, P. Zhang, Z.Y. Zhang, W.L. Qiu, D.F. Hsu, C.M. Coffran, S.M. Brown, E.M. Elango, M.D. Delacure, and F.A. Chen, "Selection and validation of differentially expressed genes in head and neck cancer," Cellular and Mol. Life Sci., Vol. 61, 2004, pp. 1372-1383.

[37] H.Y. Chuang, H.F. Liu, S. Brown, C.M. Coffran, and D.F. Hsu, "Identifying significant genes from microarray Data," in Proc. IEEE Symp. Bioinformatics and Bioengineering (BIBE'04), 2004, pp. 358-365.

[38] H.Y. Chuang, H.F. Liu, F.A. Chen, C.Y. Kao, and D.F. Hsu, "Combination methods in microarray analysis," in Proc. 7th Intl. Symp. Parallel Architectures, Algorithms and Networks (I-SPAN '04), IEEE Computer Society, 2004, pp. 625-630.

[39] D.F. Hsu and A. Palumbo, "A study of data fusion in Cayley graphs G (Sn, Pn)," in Proc. 7th Intl. Symp. Parallel Architectures, Algorithms and Networks (I-SPAN '04), IEEE Computer Society, 2004, pp. 557-562.

[40] B. Rost and C. Sander, "Prediction of protein secondary structure at better than 70% accuracy," Journal of Molecular Biology, Vol. 232, 1993, pp. 584-599.

[41] J. Jundstrom, L. Rychlewski, J. Bujnicki and A. Elofsson; "Pcons: A neural-network-based consensus predictor that improves fold recognition," Protein Science, Vol. 10, 2001, pp. 2354-2362.

[42] K.L. Lin, C. Y. Lin, C.D. Huang, H.M. Chang, C. Y. Yang, C.T. Lin, C. Y. Tang, and D. F. Hsu; "Methods of improving protein structure prediction based on HLA neural network and combinatorial fusion analysis, " *WSEAS Trans. Information Science & Applications*, Vol. 2, No. 12, 2005, pp. 2146-2153.

[43] C.S. Yu, J.Y. Wang, J.M. Yang, P.C. Lyu, C.J. Lin, and J.K. Hwang, "Fine-Grained Protein Fold Assignment by Support Vector Machines Using Generalized nPeptide Coding Schemes and Jury Voting From Multiple-Parameter Sets," *Proteins*, Vol. 50, 2003, pp.531-536.

[44] C. Bystroff and D. Baker, "Prediction of local structure in proteins using a library of sequence-structure motifs," *Journal of Molecular Biology*, Vol. 281, pp. 565–577, 1988.

[45] Y. Liu and D.L. Beveridge, "Exploratory studies of ab-initio protein structure prediction: multiple copy simulated annealing, amber energy functions, and a generalized born/solvent accessibility solvation model," *Proteins: Structure, Function, and Genetics*, Vol. 46, pp. 128-146, 2002.

[46] P. Pokarowski, A. Kolinski and J. Skolnick, "A minimal physically realistic protein-like lattice model: Designing an energy landscape that ensures all-or-none folding to a unique native state," *Biophys J.*, Vol. 84, pp. 1518–1526, 2003.

[47] G. Chikenji, Y. Fujitsuka and S. Takada, "A reversible fragment assembly method for de novo protein structure prediction," *J. Chem. Phys*, Vol. 119, pp. 6895-6903, 2003.

[48] D. Kihara and J. Skolnick, "The PDB is a covering set of small protein structures," *Journal of Molecular Biology*, Vol. 334, pp. 793-802, 2003.

[49] Y. Zhang and J. Skolnick, "The protein structure prediction problem could be solved using the current PDB library," *Proc. Natl. Acad. Sci.*, USA, Vol. 102, pp. 1029-1034, 2005.

[50] R. Kolodny and M. Levitt, "Protein decoy assembly using short fragments under geometric constraints," Biopolymers, Vol. 68, pp. 278-285, 2003.

[51] B. H. Park and M. Levitt, "The Complexity and accuracy of discrete state models of protein structure," Journal of Molecular Biology, Vol. 249, pp. 493-507, 1995.

[52] S. Anishetty, G. Pennathur and R. Anishetty, "Tripeptide analysis of protein structures," BMC Structural Biology, Vol. 2, No. 9, 2002.

[53] C. Benros, A. G. de Brevern, C. Etchebest and S. Hazout, "Assessing a novel approach for predicting local 3D protein structures from sequence," Proteins: Structure, Function, and Genetics, Vol. 62, Issue 4, pp.865-880, 2006.

[54] A.G. de Brevern and S. Hazout, "Hybrid protein model for optimally defining 3D protein structure fragments," Bioinformatics, Vol. 19, No. 3, pp. 345-353, 2003.

[55] A. G. de Brevern, C. Etchebest and S. Hazout, "Bayesian probabilistic approach for prediction backbone structures in terms of protein blocks," Proteins, Vol. 41, pp. 271-287, 2000.

[56] W. Kabsch, "A solution for the best rotation to relate two sets of vectors," Acta Crystallogr., Vol. B32, pp. 922-923, 1976.

[57] W. Kabsch, "A discussion of the solution for the best rotation to relate two sets of vectors," Acta Crystallogr., Vol. A34, pp. 828–829, 1978.