8-2-2006

Clustering System and Clustering Support Vector Machine for Local Protein Structure Prediction

Wei Zhong

CLUSTERING SYSTEM AND CLUSTERING SUPPORT VECTOR MACHINE FOR LOCAL PROTEIN STRUCTURE PREDICTION

by

Wei Zhong

Under the Direction of Yi Pan

ABSTRACT

Protein tertiary structure plays a very important role in determining its possible functional sites and chemical interactions with other related proteins. Experimental methods to determine protein structure are time consuming and expensive. As a result, the gap between protein sequence and its structure has widened substantially due to the high throughput sequencing techniques. Problems of experimental methods motivate us to develop the computational algorithms for protein structure prediction.

In this work, the clustering system is used to predict local protein structure. At first, recurring sequence clusters are explored with an improved K-means clustering algorithm. Carefully constructed sequence clusters are used to predict local protein structure. After obtaining the sequence clusters and motifs, we study how sequence variation for sequence clusters may influence its structural similarity.

Analysis of the relationship between sequence variation and structural similarity for sequence clusters shows that sequence clusters with tight sequence variation have high structural similarity and sequence clusters with wide sequence variation have poor structural similarity. Based on above knowledge, the established clustering system is used to predict the tertiary structure for local sequence segments. Test results indicate that highest quality clusters can give highly reliable prediction results and high quality clusters can give reliable prediction results.

In order to improve the performance of the clustering system for local protein structure prediction, a novel computational model called Clustering Support Vector Machines (CSVMs) is proposed. In our previous work, the sequence-to-structure relationship with the K-means algorithm has been explored by the conventional K-means algorithm. The K-means clustering algorithm may not capture nonlinear sequence-to-structure relationship effectively. As a result, we consider using Support Vector Machine (SVM) to capture the nonlinear sequence-to-structure relationship. However, SVM is not favorable for huge datasets including millions of samples. Therefore, we propose a novel computational model called CSVMs. Taking advantage of both the theory of granular computing and advanced statistical learning methodology, CSVMs are built specifically for each information granule partitioned intelligently by the clustering algorithm. Compared with the clustering system introduced previously, our experimental results show that accuracy for local structure prediction has been improved noticeably when CSVMs are applied.

INDEX WORDS: K-means clustering algorithm, PISCES (Protein Sequence Culling Server), HSSP (Homology-Derived Secondary Structure of Proteins), sequence motif, hydrophobicity index, evolutionary distance, PDB (Protein Data Bank), SVM (Support Vector Machine), protein structure prediction, granular computing.

CLUSTERING SYSTEM AND CLUSTERING SUPPORT VECTOR MACHINE FOR LOCAL PROTEIN STRUCTURE PREDICTION

by

WEI ZHONG

A Dissertation Submitted in Partial Fulfillment of Requirements for the Degree of

Doctor of Philosophy

In the College of Arts and Sciences

Georgia State University

2006

CLUSTERING SYSTEM AND CLUSTERING SUPPORT VECTOR MACHINE

FOR LOCAL PROTEIN STRUCTURE PREDICTION

by

WEI ZHONG

Major Professor: Yi Pan
Committee: Phang C. Tai
Robert Harrsion
Martin D. Fraser

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

August 2006

ACKNOWLEDEMENTS

The dissertation would not have been possible without the help of so many people. I would like to take this opportunity to express my deep appreciation to all those who helped me in this hard but extremely rewarding process.

First, I would like to thank my advisor, Professor Yi Pan, for all his help, advice, support, guidance, and patience. Whenever I have difficulties and problems, Dr. Pan always encourages me to make harder efforts so that I can overcome those problems. Dr. Pan provides me so many valuable advices about how to conduct research and choose my career. Without his help, I could not make rapid progress in my research and dissertation writing.

I am grateful to Dr. Robert Harrison, Dr. Phang. C. Tai, and Dr. Martin D. Fraser for serving on my Ph.D committee, and for their time and cooperation in reviewing this work. Dr. Harrison provides me insights into how to cluster the sequence segments and how to express my ideas clearly in my paper. I would like to thank Dr. Fraser for giving me many valuable suggestions about statistical techniques for SVM learning. I would like to thank Dr. Tai for providing me a lot of biological knowledge so that I can combine the computational methods with biological experiments smoothly. I would like to thank Professor Roland L. Dunbrack for providing the data set from PISCES. This research was supported in part by the U.S. National Institutes of Health under grants R01 GM34766-17S1, and P20 GM065762-01A1, and the U.S. National Science Foundation under grants ECS-0196569, and ECS-0334813. I am also supported by a Georgia State University Molecular Basis of Disease Program Fellowship.

I would like to thank Dr. Raj Sunderraman for his great patience and support during my job search and Ph.D. study. The more important thanks are reserved for the last. Many thanks to my parents, Zhong Shunlong and Wei Shufang, for their constant support, concern, and motivation. My parents give many important advices about how to lead a successful life during my Ph.D. study.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

Clustering Support Vector Machines	CSVMs
Support Vector Machine	SVM
Nuclear Magnetic Resonance	NMR
The Basic Local Alignment Search Tool	BLAST
Protein Structure Prediction and Evaluation Computer Toolkit	PROSPECT
Critical Assessment of Techniques for Protein Structure Prediction	CASP
Position Specific Iterative BLAST	PSI-BLAST
Position Specific Scoring Matrix	PSSM
Structural Classification of Proteins	SCOP
Root Mean Squared Deviation	RMSD
Hidden Markov Models	HMM
Protein Sequence Culling Server	PISCES
Homology-derived Secondary Structure of Proteins	HSSP
Protein Data Bank	PDB
Simultaneous Multithreading	SMT
Average Distance Matrix	ADM
Sequential Minimal Optimization	SMO
Distance Matrix Root Mean Square Deviation	dmRMSD
Torsion Angle Root Mean Square Deviation	taRMSD

Chapter 1 Introduction

1.1 Research Motivations and Contributions

1.1.1 Local Protein Structure Prediction

Proteins are polymers of amino acids connected by formation of covalent peptide bonds. Proteins have four levels of structures including primary structure, secondary structure, tertiary structure and quaternary structure. Based on hydrogen bonding interactions between adjacent amino acid residues, the polypeptide chain can arrange itself into secondary structure. The polypeptide chains of protein molecules fold into the native structure. Multiple interacting polypeptide chains of characteristics tertiary structure develop into protein quaternary structure.

Protein structure can be determined experimentally by X-ray crystallography, Nuclear Magnetic Resonance (NMR) and electron microscopy. When X-ray crystallography is applied, crystallisation of proteins is a very difficult task. Compared to X-ray crystallography, experiments related to NMR are carried out in solution rather than a crystal lattice. However, NMR can only be applicable to determine structures of small and mediums-sized molecules due to limitation of the principle that make NMR possible.

Knowledge about protein functions can be used to infer how the protein interacts with other molecules. The protein functions are largely determined by their structures. As a result, understanding protein structures is a very important task. Determination of protein structure by experimental methods is a long and tedious process. Difficulties of determining protein

structures experimentally require us to predict protein structures using computational methods. Comparative homology modeling, threading, and Ab Initio method are three major methods for protein structure prediction. The classification of these three major methods is based on how each method utilizes the available resources in the current database.

Comparative homology modeling produces the best prediction results so far. The tertiary structure and functions are highly conserved during the evolutionary process. As a result, protein sequences with high sequence similarity usually share similar structures. The prediction accuracy of homology modeling depends on whether protein sequences in the protein data bank that have high sequence similarity with target protein sequences can be found. Sequence alignment algorithms are used to find protein sequences sharing high similarity with target sequences whose structure to be predicted. Based on sequence alignment algorithms, the aligned residuals of the structure templates from protein sequences sharing high similarity with target sequences are used to construct the structural model. In this process, the quality of sequence alignment algorithms is the key factor to determine whether suitable structural templates can be selected and how well the target protein can be aligned with structural templates.

For the comparative homology modeling, local sequence alignment is used to find out segments of the protein sequences with high similarity. Local sequence alignment includes pairwise alignment and profile-based alignment. Profile-based methods perform much better than the pairwise comparison such as the Basic Local Alignment Search Tool (BLAST) when sequence similarity is less than 30%.

If sequence alignment algorithms cannot find correct folds for the target sequence, threading or fold recognition can be utilized to provide the correct folds to the target sequence. Based on the concept that only a small number of distinct protein folds exist for protein families, a library

of representative local structures is scanned in order to find structure analogs to protein sequences. After the library is set up, the energy function is used to select the suitable library entries serving as the templates for target sequences. Protein Structure Prediction and Evaluation Computer Toolkit (PROSPECT) is one of the best threading programs in the Critical Assessment of Techniques for Protein Structure Prediction (CASP) competition (Xu et. al., 2001). The threading methods are computationally expensive because each entry of the library having thousands of possible folds is required to be aligned in all possible ways. The energy function used in threading methods are not sophisticated enough to find the correct protein folds.

Ab Initio methods can be used to predict protein structures from the sequence information when appropriate structure templates cannot be found. Most Ab Initio prediction methods restrict the conformation space to the reasonable size using reduced protein representation and select those energy functions related to the most important interactions responsible for protein folding in its native form.

1.1.2 Clustering System for Local Protein Structure Prediction

Recurring sequence motifs of proteins are explored with an improved K-means clustering algorithm. Information about local protein sequence motifs is very important to the analysis of biologically significant conserved regions of protein sequences. These conserved regions can potentially determine the diverse conformation and activities of proteins. Carefully constructed sequence motifs from sequence clusters are used to predict local protein structure.

PROSITE, PRINTS and PFAM are popular methods to create sequence motifs. Since sequence motifs and profiles of PROSITE, PRINTS and PFAM are developed from multiple sequence alignments, these sequence motifs and profiles only search conserved elements of

sequence alignments from the same protein family and carry little information about conserved sequence regions, which transcend protein families. Furthermore, the knowledge about the biologically important regions or residues is the precondition of finding these motifs. As a result, the discovery of sequence motifs and profiles requires intensive human intervention. While these methods to produce the popular sequence motifs require human intervention to explore the biologically significant regions of protein sequences, the clustering technique provides an automatic, unsupervised discovery process. All these advantages, in comparison to these methods to create popular sequence motifs, motivate us to develop an improved K-means clustering algorithm.

Han and Baker have used the K-means clustering program to find recurring local sequence motifs for proteins (Han and Baker, 1995; Han and Baker, 1996). In their work, a set of initial points for cluster centers is chosen randomly (Han and Baker, 1995). Since the performance of K-means clustering is very sensitive to initial point selection (Jain, Murty and Flynn, 1999), their technique may not yield satisfactory results. To overcome potential problems of random initialization, the new greedy initialization method tries to choose suitable initial points so that final partitions can represent the underlying distribution of the data samples more consistently and accurately (Zhong et.al, 2004b). Each initial point is represented by one local sequence segment. In the new initialization method, the clustering algorithm will only be performed for several iterations during each run. After each run, initial points, which can be used to form the cluster with good structural similarity, are chosen and their evolutionary distance is checked against that of all points already selected in the initialization array. If the minimum evolutionary distance of new points is greater than the specified distance, these points will be added to the initialization array. Satisfaction of the minimum evolutionary distance can guarantee that each

newly selected point will be well separated from all the existing points in the initialization array and will potentially belong to different natural clusters**.** This process will be repeated several times until the specified number of points is chosen. After this procedure, these carefully selected points can be used as the initial centers for the K-means clustering algorithm**.**

Analysis of the clustering process of the traditional clustering algorithm reveals that some of the initial points are very close to each other, creating strong interferences with each other. Strong interferences among initial points will affect final partitioning negatively. The results of our improved K-means algorithm show the average percentage of sequence segments belonging to clusters with structural similarity greater than 60% steadily improves with increasing minimum evolutionary distances among initial points. This improved percentage results from decreased interferences among initial points when the evolutionary distances among initial points are increased. Comparison between sequences motifs obtained by both algorithms suggests that the improved K-means clustering algorithm may discover some relatively weak and subtle sequence motifs. These motifs are undetectable by the traditional K-means algorithm because random selection of points may choose two starting points that are within one natural cluster. For example, some of the weak amphipathic helices and sheets discovered by the improved K-means algorithm have not been reported in the literature. In addition, the number of repeated substitution patterns of sequence motifs found by the traditional K-means algorithms is less than that of the improved K-means algorithms.

Our results reveal much more detailed hydrophobicity patterns for helices, sheets and coils than the previous study (Han and Baker, 1995). These elaborate hydrophobicity patterns are supported by various biochemical experiments. Increased information about hydrophobicity patterns associated with these sequence motifs can expand our knowledge of how proteins fold

and how proteins interact with each other. Furthermore, the analysis of discovered sequence motifs shows that some elaborate and subtle sequence patterns such as Pattern 1, 9, 22 have never been reported in previous works. Especially, increased number of repeated substitution patterns reported in this study may provide additionally strong evidences for structurally conservative substitutions during the evolutionary process for protein families.

The sequence motifs discovered in this study indicate conserved residues that are structurally and functionally important across protein families because protein sequences used in this study share less than 25% sequence identities. These important features from our sequence motifs may help to compensate for some of the weak points of those created by PROSITE, PRINTS, PFAM and BLOCKS (Attwood et al., 2002; Henikoff, Henikoff and Pietrokovski, 1999; Sonnhammer et.al., 1998). Our sequence motifs may reflect general structural or functional characteristics shared by different protein families while sequence motifs from PROSITE, PRINTS, PFAM and BLOCKS represent structural or functional constraints specific to a particular protein family. Due to the high throughput sequencing techniques, the number of known protein sequences has increased rapidly in recent years. However, information about functionally significant regions of these new proteins may not be available. As a result, automatic discovery of biologically important sequence motifs in this study is a much more powerful tool to explore underlying correlations between protein sequences, structures and functions than other methods requiring guidance from existing scientific results.

In our study, the cluster number of 800 is chosen empirically. However, 800 may not be the optimal cluster number. Therefore, the improved K-means algorithm will be run several times with different values of k in order to discover the most suitable number of clusters. With the information about the optimal cluster number, clustering results may be potentially closest to

underlying distribution patterns of the sample space. However, the time spent searching for the good initial points grows substantially when the minimum evolutionary distance and structural similarity threshold are increased. For example, it will take 18 days to obtain appropriate initial points with the distance threshold of 1500 when the sample size is very large. Due to the time and processing power constraints, the search for the optimal cluster number has not been completed. The long searching time for initial points motivates us to implement the parallel K-means algorithm in order to reduce the searching time for suitable initial points to one to two days. The parallelization of the improved K-means algorithm will make exploration of the optimal cluster number possible. We predict that the performance gains for the improved K-means algorithm will be increased further after the optimal cluster number is found. As a result, Pthread and OpenMP are employed to parallelize K-means clustering algorithm in the Hyper-Threading enabled Intel architecture. Speedup for 16 Pthreads is 4.3 and speedup for 16 OpenMP threads is 4 in the 4 processors shared memory architecture. With the new parallel K-means algorithm, K-means clustering can be performed for multiple times in reasonable amount of time. Our research also shows that Hyper-Threading technology for Intel architecture is efficient for this parallel biological algorithm.

After we propose an improved K-means clustering algorithm to discover the sequence clusters and sequence motifs automatically and to implement the parallel K-means clustering algorithm, we want to discuss how sequence variation for sequence clusters may influence its structural similarity. Analysis of the relationship between the sequence variation and corresponding structural variation for sequence clusters is one of open questions for protein structure and sequence analysis (Rahman and Zomaya, 2005). Some researchers have evaluated the structural variation for sequence clusters. Kasuya and Thornton (1999) and Jonassen et al.

(1999) have used cRMSD to analyze structural variation for sequence motifs. Bystroff and Baker (1998) have used the K-means clustering algorithm to find sequence clusters and to assess structural variation for these sequence clusters. Bystroff and Baker incorporated structural information during the clustering process (1998). As a result, final sequence clusters are contaminated by usage of structural information during the clustering process. Our implementation of the K-means clustering is significantly different from Bystroff's work (1998) because we only use recurrent clusters and do not include structural information in the clustering process. To the best of our knowledge, no researchers have conducted in-depth analysis of the relationship between sequence variation and corresponding structural variation for sequence clusters (Zhong et.al, 2005a).

This work focuses on systematic and detailed analysis of the relationship between sequence variation and corresponding structural variation for sequence clusters. Understanding this relationship is very important to improve the quality of local sequence alignment and low homology protein folding. Sequence clusters with tight sequence variation can be used to establish structural templates for low homology protein folding. Frequency profile of sequence clusters with tight sequence variation also can be used to find sequence segments with similar local structure in the local sequence alignment algorithm.

Since the average of relative entropy values for all positions of frequency profiles cannot determine the sequence variation for sequence clusters, we use the number of important position to define the sequence variation for sequence clusters. If the relative entropy in the specified position of the frequency profiles is greater than 0.2, this position is defined as the important position for frequency profiles. Our statistics indicate that an average of five amino acids occupy 60% of the frequency space if the relative entropy in that position of the frequency profiles is

greater than 0.2. Statistically, each of twenty amino acids may occur with the frequency of 5%. Therefore, five amino acids may occupy 25% of the frequency space. As a result, the distribution of amino acids is highly disproportionate in the important positions.

The number of important positions is used to indicate the extent of sequence variation for sequence clusters. Increased number of important positions in the frequency profiles reflects more positions in the frequency profiles have highly disproportionate distribution of 20 amino acids. As a result, sequence variation for sequence clusters is more compact. In contrast, relatively small number of important position indicates the sequence variation for sequence clusters is wide. Our results indicate that defining sequence variation for sequence clusters by the number of important position is more effective in distinguishing the sequence clusters with high structural variation and low structural variation.

The sequence variation and structural variation of sequence clusters having sequence segments with the specified length are analyzed separately. The length of sequence segments ranges from 5 to 15 in our study. Sequence clusters having sequence segments with different lengths show the similar relationship between sequence variation and structure variation for sequence clusters. Due to limitation of space, we focus on the sequence cluster containing sequence segments with the length of nine. All the results shown in the following are related to the sequence clusters having sequence segments with the length of nine.

Analysis of our results reveals that on average, the number of important positions for clusters with low structural variation is greater than the number of important positions for clusters with high structural variation. Low structural variation for sequence clusters indicates that structural variation is compact. A large number of important positions indicate that sequence variation for sequence clusters is tight. In other words, our results indicate the important pattern that sequence

clusters with tight sequence variation tend to have tight structural variation and sequence clusters with wide sequence variation tend to have wide structural variation.

After we explain the improved K-means algorithm for sequence motif discovery and how sequence variation for sequence clusters may influence its structural similarity, the clustering system is developed for local protein structure prediction. Our preliminary results show that the sequence segments with the length of nine are long enough to have some structural features and are short enough to have a statistically significant number of samples. It is clear that other segment lengths are important and the analysis presented here can be applied to them as well. Due to huge amount of computation, we plan to analyze the sequence segments from the length ranging from 5 to 15 in the next step. Average distance matrix, representative torsion angle and representative secondary structure are the representative structure of each cluster.

The frequency profile for a given sequence segment is compared with the centroid of the each cluster in order to calculate distance score. A smaller distance score shows that the frequency profile of the given sequence segment is closer to the centroid for a given cluster. The reliability score of a given sequence segment for a cluster is determined by the sum of the frequency of the matched amino acid in the corresponding position of the average frequency profile of a cluster. The distance score of each cluster for a given sequence segment is calculated in order to filter out some less significant cluster. If the difference of the cluster's distance score and the smallest distance score is within 100, this cluster is selected. Other clusters are discarded since they are less significant. The cluster with the highest reliability score among the selected clusters finally functions to predict the structure of this sequence segment. Our results indicate that clusters with high quality provide the reliable prediction results and clusters with average

quality produces high quality results. Special caution need be taken against prediction results by the bad cluster group.

1.1.3 Clustering Support Vector Machine for Local Protein Structure Prediction

The central ideas of support vector machines are to map the input space into another higher dimensional feature space using the kernels function and to build an optimal hyperplane in that feature space (Vapnik, 1998). One of important questions is that how we can build the hyperplane that has strong generalization capability in the high dimensional feature space. The second question is that how we can avoid the "curse of dimensionality" in this high dimensional feature space. The Mercer's Theorem helps us avoid mapping the input space into another higher dimensional space explicitly. Mercer's theorem indicates that any kernel function satisfying Mercer's condition can calculate the inner product of two vectors in some high dimensional Hilbert space. Based on Mercer theorem, the high-dimensional feature space need not be considered directly during the process of finding the optimal hyperplane. Instead, the inner products between support vectors and the vectors in the feature space can be calculated.

SVM has two layers. In the first layer, input vectors are implicitly transformed and each inner product between the input vector and support vectors are calculated based on the kernel function. In the second layer, the linear decision function is built in the high dimensional feature space. The best SV machine with the smallest expected risks has smallest VC dimension.

SVMs are based on the idea of mapping data points to a high dimensional feature space where a separating hyperplane can be found. SVMs are searching the optimal separating hyper-plane by solving a convex quadratic programming (QP). The typical running time for the convex quadratic programming is Ω (m^2) for the training set with m samples. The convex quadratic programming is NP-complete in the worst case (Vavasis, 1991). Therefore, SVMs are not

favorable for a large dataset (Chang and Lin, 2001). Our dataset contains a half millions samples. Experimental results show that training of SVM for a half millions samples is not complete after one month on the "poweredge6600 server" with four processors from Dell®.

Many algorithms and implementation techniques have been developed to enhance SVMs in order to increase their training performance with large data sets. The most well-known techniques include chunking (Vapnik, 1998), Osuna's decomposition method (Osuna, Freund, and Girosi, 1997), Sequential Minimal Optimization (SMO) (Platt, 1999) and boosting algorithms (Pavlov, Mao and Dom, 2000). The success of these methods depends on dividing the original quadratic programming (QP) problem into a series of smaller computational problems in order to reduce the size of each QP problem. Although these algorithms accelerate the training process, these algorithms do not scale well with the size of the training data.

The second class of algorithms tries to speed up the training process by reducing the number of training data. Since some data points such as the support vectors are more important to determine the optimal solution, these algorithms provide SVMs with high quality data points during the training process. Random Selection (Balcazar, Dai and Watanabe, 2001) and clustering analysis (Yu, Yang, and Han, 2000) are representatives of these algorithms. Their algorithms are highly scalable for the large data set while the performance of training depends greatly on the selection of training samples.

In order to solve the problems related to large sample training, Clustering Support Vector Machines are proposed in this work. Understanding protein sequence-to-structure relationship is one of the most important tasks of current bioinformatics research. The knowledge of correspondence between the protein sequence and its structure can play very important role in protein structure prediction (Rahman and Zomaya, 2005). Han and Baker have used the K-means

clustering algorithm to explore protein sequence-to-structure relationship. Protein sequences are represented with frequency profiles. With the K-means clustering algorithm, high quality sequence clusters have been produced (Han and Baker, 1996). They have used these high quality sequence clusters to predict the backbone torsion angles for local protein structure (Bystroff and Baker, 1998). In their work and our previous works, the K-means clustering algorithm is essential to understand how protein sequences correspond to local 3D protein structures. However, the conventional clustering algorithms such as the K-means and K-nearest neighbor algorithm assume that the distance between data points can be calculated with exact precision. When this distance function is not well characterized, the clustering algorithm may not reveal the sequence-to-structure relationship effectively. As a result, some of clusters provide poor correspondence between protein sequences and their structures.

SVM can handle the nonlinear classification by implicitly mapping input samples from the input feature space into another high dimensional feature space with the nonlinear kernel function. Therefore, SVM may be more effective to reveal the nonlinear sequence-to-structure relationship than K-means clustering does. The superior performance for non-linear classification inspires us to explore the relationship between the protein sequence and its structure with SVM.

Training SVM over the whole feature space containing almost half million data samples takes a long time. Furthermore, each subspace of the whole feature space corresponds to different local 3D structures in our application. As a result, construction of one SVM for the whole feature space cannot take advantage of the strong generalization power of SVM efficiently. The disadvantage of building one SVM over the whole feature space motivates us to consider the theory of granular computing.

Granular computing decomposes information in the form of some aggregates such as subsets, classes, and clusters of a universe and then solves the targeted problems in each granule (Yao, 2004). Granular construction and computing are two major tasks of granular computing (Yao, 2005). Granular computing conceptualizes the whole feature space at different granularities and switch among these granularities (Yao, 2004). With the principles of divide-and-conquer, granular computing breaks up the complex problems into smaller and computationally simpler problems and focuses on each small problem by omitting unnecessary and irrelevant information. As a result, granular computing can increase intelligence and flexibility of data mining algorithms.

To combine the theory of granular computing and principles of the statistical learning algorithms, we propose a new computational model called Clustering Support Vector Machines (CSVMs) in our work. In this new computational model, one SVM is built for each information granule defined by sequence clusters created by the clustering algorithm. CSVMs are modeled to learn the nonlinear relationship between protein sequences and their structures in each cluster. SVM is not favorable for large amount of data samples. However, CSVMs can be easily parallelized to speed up the modeling process. After gaining the knowledge about the sequence to structure relationship, CSVMs are used to predict distance matrices, torsion angles and secondary structures for backbone α-carbon atoms of protein sequence segments. Compared with the clustering system introduced previously, CSVMs can estimate how close frequency profiles of protein sequences correspond with local 3D structures by using the nonlinear kernel. Introduction of CSVMs can potentially improve the accuracy of local protein structure prediction.

CSVMs are built from information granules, which are intelligently partitioned by clustering algorithms. Intelligent partitioning by clustering algorithms provides true and natural representations of inherent data distribution of the system. Because of data partitioning, a complex classification problem is converted into multiple smaller problems so that learning tasks for each CSVM are more specific and efficient (He et al., 2006). Each CSVM can concentrate on highly related samples in each feature subspace without being distracted by noisy data from other clusters. As a result, CSVMs can potentially improve the generalization capability for classification problems.

Since granulation by K-means clustering may introduce noise and irreverent information into each granule, the machine learning techniques are required to identify the strength of correspondence between frequency profiles and 3D local structure for each sequence segment belonging to the same information granule. After learning the relationship between frequency profile distribution and 3D local structures, CSVMs can filter out potentially unreliable prediction and can select potentially reliable prediction for each granule.

Because our unpublished results reveal that the distribution patterns for frequency profiles in each cluster is quite different, the functionality and training of CSVMs is customized for each cluster belonging to different cluster groups. The CSVMs for clusters belonging to the bad cluster group are designed to identify sequence segments whose structure can be reliably predicted. The CSVMs for clusters belonging to the good cluster group are trained to filter out sequence segments whose structure cannot be reliably predicted.

Local protein structure prediction by CSVMs is based on the prediction method from the clustering algorithm. At first, the sequence segments whose structures to be predicted are assigned to a specific cluster in the cluster group by the clustering algorithm. Then CSVM

trained for this specific cluster is used to identify how close the frequency profile of this sequence segment is nonlinearly correlated to the 3D local structure of this cluster. If the sequence segment is predicted as the positive sample by CSVM, the frequency profile of this segment has the potential to be closely mapped to 3D local structure for this cluster. Consequently, the 3D local structure of this cluster can be safely assigned to this sequence segment. The method to decide the 3D local structure of each cluster can be found in Chapter 12. If the sequence segment is predicted as the negative sample by CSVM, the frequency profile of this segment does not closely corresponds to the 3D local structure for this cluster. The structure of this segment cannot be reliably predicted by this cluster. This cluster is removed from the cluster group. The cluster membership function calculating distance scores and reliability scores is used to select the next cluster from the remaining clusters of the cluster group. The previous procedure will be repeated until one SVM modeled for the selected cluster predict the given sequence segment as positive. Important knowledge about the correspondence between frequency profiles and the 3D local structure provided by CSVMs can provide the additional dependable metric of cluster membership assignment.

Average accuracy for CSVMs is over 80%, which indicates that the generalization power for CSVMs is strong enough to recognize the complicated pattern of sequence-to-structure relationships. CSVM modeled for different cluster group obtains good capability to discriminate between positive samples and negative samples. CSVMs for the bad cluster group are able to select frequency profiles of sequence segments whose structure can be reliably predicted. The recall value for CSVMs belonging to the good cluster group reaches 96%. This high value reveals that CSVMs did not misclassify frequency profiles of sequence segments whose structure can be accurately predicted. The precision value for CSVMs belonging to the good cluster group

reaches 86%. The high precision value demonstrates that CSVMs belonging to the good cluster group obtain the capability to filter out the frequency profiles of sequence segments whose structure cannot be reliably predicted.

Compared with the clustering system introduced previously, our experimental results show that accuracy for local structure prediction has been improved noticeably when CSVMs are applied.

1.2 Dissertation Organization

This dissertation has been divided into four parts. In the first part of dissertation, I discuss how protein structures are represented and why protein structure prediction is important. The first part covers Chapter 2. In the second part of dissertation, I discuss the new improved K-means clustering for sequence cluster and motif discovery. Then I explain how sequence variation for sequence clusters may influence its structural similarity. Based on the above information, the clustering system is developed in order to carry out local protein structure prediction. The second part expands from Chapter 3 to Chaper 8.

The third part of the dissertation discusses the new clustering support machine to perform local protein structure prediction since the clustering system used in the second part may not capture non-linear sequence to structure relationship effectively. The third part of the dissertation also explains the conclusions and future work. The third part covers Chapter 9. The fourth part of the dissertation will provide the conclusions and future work. The fourth part covers Chapter 10.

In Chapter 2, four levels of protein structure are explained first. Then how protein structure can be experimentally determined is introduced. In the third part of this chapter, three major computational methods to predict protein structure are discussed in details.

In Chapter 3, an improved K-means clustering algorithm is introduced in order to explore recurring sequence motifs of proteins. Information about local protein sequence motifs is very important to the analysis of biologically significant conserved regions of protein sequences. This chapter has been divided into five sections. First, the major motif discovery methods are discussed. Then, the major characteristics of the traditional and improved K-means algorithms are compared. In section 3.3, the experimental setup is explained. In section 3.4, experimental results are presented to show that the improved K-means algorithm is better than the traditional K-means algorithm and to give evidence that our research find some previously undiscovered sequence motifs. In section 3.5, our research is compared to other state-of-art approaches in order to emphasize the advantages of our research.

The long searching time for initial points motivates us to implement the parallel K-means algorithm in order to reduce the searching time for suitable initial points to one to two days. In Chapter 4, the parallel K-means algorithm is introduced. The parallelization of the improved K-means algorithm will make exploration of the optimal cluster number possible. We predict that the performance gains for the improved K-means algorithm will be increased further after the optimal cluster number is found. In this chapter, two important parallelization techniques for the K-means clustering algorithm are discussed. Then programming environment and implementation details are explained. Finally, experimental results for speedup values are presented.

In Chapter 5, we want to discuss how sequence variation for sequence clusters may influence its structural similarity. How sequence variation for sequence clusters may influence its structural similarity is one of the most important tasks of current bioinformatics research. In this chapter, previous studies for sequence and structural variation of sequence clusters are reviewed

first. Then recurrent clustering, data set and generation of sequence segments are introduced. Evaluation of sequence variation and structural similarity is discussed in detail. Finally, results of analysis about the relationship between sequence variation and structural variation are given.

In Chapter 3 and 5, we have discussed the improved K-means algorithm for sequence motif discovery and how sequence variation for sequence clusters may influence its structural similarity. Based on above knowledge, the clustering system is developed for local protein structure prediction in the Chapter 6. In this chapter, how to cluster sequence segments into clusters is explained first. Then the method to calculate the representative structure for each cluster is explained. Distance score and reliability score to decide the cluster membership is discussed. The performance evaluation and experimental results are explained in the last part of this chapter.

In Chapter 7, Support Vector Machines will be explained in details. Support Vector Machines are a new generation of learning machines, which have been successfully applied to a wide variety of application domains (Cristianini and Shawe Taylor, 2000) including bioinformatics (Schoelkopf, Tsuda and Vert, 2000). Construction of optimal hyperplane that can separate samples belonging to the first class from samples belonging to the second class with the maximal margin is the essential task of SVM. In this chapter, the concept of optimal hyperplane and optimization problems to construct optimal hyperplane in the linearly separable case and in the linearly nonseparable case will be discussed first. Then the expected risk bounds are evaluated to assess the effectiveness of support vector machines. In addition, the quadratic optimization and linear optimization method to build SVMs are discussed. SVM Kernels play key roles in calculating the inner products between support vectors and the vectors implicitly in the high dimensional feature space, several important SVM kernels are introduced in this section. In real

world, we need solve the multiclassification problem besides two-class classification. Multiple classifications for SVM are also explained.

SVMs are not favorable for a large dataset (Chang and Lin, 2001). In Chapter 8, many algorithms and implementation techniques developed to enhance SVMs in order to increase their training performance with large data sets is introduced. In this chapter, the algorithms dividing the original quadratic programming (QP) problem into a series of smaller computational problems is discussed first. Then the second class of algorithms trying to speed up the training process by reducing the number of training data is explained.

In Chapter 9, the Clustering Support Vector Machines is introduced for protein local structure prediction. In our previous approaches, the conventional clustering algorithms are used to capture the sequence-to-structure relationship. The cluster membership function defined by conventional clustering algorithms may not reveal the complex nonlinear relationship adequately. As a result, the new computational model called Clustering Support Vector Machines is proposed to carry out local protein structure prediction. In the section 9.1, previous researches are reviewed. In the section 9.2, the advantages of granular computing and SVM are introduced. A new computational model called Clustering Support Vector Machines is also discussed in details. In the section 9.3, the training set, the testing set and accuracy definition are explained. In the section 9.4, the experimental results and analysis are given. Finally, the conclusion and the future work are presented.

In Chapter 10, the conclusions and future work is given. In this chapter, the new cluster membership function, kernel selection feature selection is proposed in order to improve the accuracy of SVM. Furthermore, I propose studying the relationship among clusters and comparing the performance of parallel SVM and CSVMs.

Chapter 2 Protein Structure Prediction

Protein tertiary structure plays a very important role in determining its possible functional sites and chemical interactions with other related proteins. Prior knowledge about protein three-dimensional structure is very helpful for protein engineering and drug design. For example, if the structure of a certain protein that causes a disease is determined, a chemical reaction related to this protein can be found out to facilitate drug research. Researchers try to determine the tertiary structure of proteins using X-ray crystallography and Nuclear Magnetic Resonance (NMR). Both methods are time consuming and expensive. Sometimes researchers fail to find out the three-dimensional coordinates of an amino acid using X-ray crystallography and NMR. As a result, the gap between protein sequence and its structure has widened substantially due to the high throughput sequencing techniques. The growing gap increases the significance of predicting the protein tertiary structure. Prediction of protein local structure is an intermediary step to explore its tertiary structure. Many biochemical tests suggest that a sequence determines conformation completely because all the information, which is necessary to specify protein interaction sites with other molecules, is embedded into its amino acid sequence. This close relationship between a sequence and a structure forms the theoretical basis for protein structure prediction.

In this chapter, four levels of protein structure are explained first. Then how protein structure can be experimentally determined is introduced. In the third part of this chapter, three major computational methods to predict protein structure are discussed in details.

2.1 Protein Structure Representations and Protein Structure Determination

Proteins are polymers of amino acids connected by formation of covalent peptide bonds. Proteins have four levels of structures including primary structure, secondary structure, tertiary structure and quaternary structure. The primary structure is the amino acid sequence. Based on hydrogen bonding interactions between adjacent amino acid residues, the polypeptide chain can arrange itself into helix, coils or sheets. A tertiary structure of protein is generated after the polypeptide chains of protein molecules fold into the native form. Multiple interacting polypeptide chains of characteristics tertiary structure develop into protein quaternary structure.

Protein structure can be determined experimentally by X-ray crystallography, Nuclear Magnetic Resonance (NMR) and electron microscopy. When X-ray crystallography is applied, crystallisation of proteins is a very difficult task. Crystals can be formed by slowly precipitating proteins under conditions keeping its native conformation. Crystallisation is a long and tedious process. Compared to X-ray crystallography, experiments related to NMR are carried out in solution rather than a crystal lattice. However, NMR can only be applicable to determine structures of small and mediums-sized molecules due to limitation of the principle that make NMR possible.

Protein functions play important roles in deciding how the protein interacts with other molecules. The protein functions are largely decided by their structures. As a result, understanding protein structures becomes one of central tasks of biological research. Protein structures can be determined by experimental methods introduced previously. However, determination of protein structure is the long and tedious process. Sometime researchers may fail to determine protein structures especially transmembrane proteins. Difficulties of determining protein structures experimentally motivate us to predict protein structures using computational

methods. Comparative homology modeling, threading, and Ab Initio method are three major methods for protein structure prediction. The classification of these three major methods is based on how each method utilizes the available resources in the current database.

2.2 Comparative Homology Modeling

Comparative homology modeling produces the best prediction results so far. During the evolutionary process, amino acids may be added, deleted or substituted in some positions of protein sequences. However, the tertiary structure and functions are highly conserved in this process. As a result, protein sequences with high sequence similarity usually share similar structures. In contrast, protein structures with high structural similarity may not share high sequence similarity. The comparative homology modeling is looking for structurally known proteins, which share similar structures with target proteins whose structures to be predicted. The prediction accuracy of homology modeling depends on whether protein sequences in the protein data bank that share high sequence similarity with target protein sequences can be found.

Homology modeling need take four steps to predict protein structures. In this first step, several suitable structural templates from the known protein structure database are selected. In the second step, the target sequence whose structure to be predicted is aligned to the structural templates. In the third step, the backbone structure, including helix, coils, sheets and other areas that are significantly different from the template structure is built. In the fourth step, the side-chains in the protein backbone structure are placed. Sequence alignment algorithms are used to find protein sequences sharing high similarity with target sequences. Based on sequence alignment algorithms, the aligned residuals of the structure templates from protein sequences sharing high similarity with target sequences are used to construct the structural model. In this process, the quality of sequence alignment algorithms is the key factor to determine whether

suitable structural templates can be selected and how well the target protein can be aligned with structural templates. The high quality alignment between the target protein and structural templates will increase the prediction accuracy of comparative homology modeling. As a result, increasing the quality of sequence alignment algorithms is a very important research issue for homology modeling. The quality of sequence alignment algorithms is evaluated by its capability to find remote homologues and to align the target sequences to other related sequences reasonably.

For the comparative homology modeling, local sequence alignment is used to find out segments of the protein sequences with high similarity. Local sequence alignment includes pairwise alignment and profile-based alignment. The Basic Local Alignment Search Tool (BLAST) is one of widely used pairwise alignment (Altschul, 1990). The BLAST can detect sequence similarity greater than 30%. In order to increase the capability for alignment algorithms to detect remote homologues, Position Specific Iterative BLAST (PSI-BLAST) is proposed (Altschul et. al., 1997). The PSI-BLAST iteratively searches the database until no new hits can be found. Since the evolutionary information about the whole family is embedded into Position Specific Scoring Matrix (PSSM), the PSI-BLAST has the capability to find protein sequences with low similarity. In order to further improve sensitivity of sequence alignment algorithms, the profile-profile based sequence alignment algorithm (Koehl and Levitt, 2002) is proposed. Profile-profile methods can find sequences with similarity less than 20%.

Based on sequence alignment, the residuals of the structure templates are aligned to the target sequences in order to construct structural models. The aligned residues are generally different from that of structure-structure alignment especially when the sequence similarity is low. The quality of sequence alignment algorithm can be evaluated based on comparison between

sequence-sequence alignment and structure-structure alignment. In order to assess how well sequence alignment algorithms can effectively align difference sequences, Sander et. al (2000) compared several sequence alignment algorithms with structural alignment algorithm such as Structural Classification of Proteins (SCOP) (Murzin et. al., 1995). Profile-based methods such PSI-BLAST and profile-to-profile performs much better than the pairwise comparison such as BLAST when sequence similarity is less than 30%.

The performance of comparative homology modeling is strongly affected by the degree of similarity between the target sequence and template sequences. If two protein sequences share sequence similarity greater than 50%, Root Mean Squared Deviation (RMSD) of the alignable sections between two sequences is usually less than 1 Å (Gerstein and Levitt, 1998). If the sequence similarity between two sequences is between 20% and 30%, most of protein sequences will have different structures. If the template from the database with known structure can be found in this case, RMSD of the alignable sections between two sequences is usually greater than 2 Å (Chung and Subbiah, 1996). If the sequence similarity between two sequences is between 8% and 10%, RMSD of the alignable sections between two sequences is as large as 6 Å. The big RMSD errors are largely created by the misalignment of two sequences.

2.3 Threading or Fold Recognition

For some evolutionary remotely related proteins, suitable template sequences cannot be found even with the most effective sequence alignment algorithm. On the same time, structural alignment algorithms can discover homologous protein sequence pairs with sequence similarity less than 10% (Rost, 1997). If sequence alignment algorithms cannot find correct folds for the target sequence, threading or fold recognition can be utilized to provide the correct folds to the target sequence.

Based on the concept that only a small number of distinct protein folds exist for protein families, a library of representative local structures is scanned in order to find structure analogs to protein sequences. After the library is set up, the energy function is used to select the suitable library entries serving as the templates for target sequences. The threading method has been divided into four categories. In the first category, the energy function is based on the environmental information of each residue in the structure and dynamic programming is used to evaluate the quality of alignment (Bowie, luthyan and Eisenberg, 1991). In the second category, the energy function takes advantages of statistically derived pairwise interaction potentials (Sippl, 1990) between the target sequence and library entries (Jones et. al., 1992). In the third category, no energy function is used. In the third category, the target sequence and library entries are encoded into strings in order to carry out sequence-structure alignment. This sequence-structure alignment uses the prediction results for secondary structure and accessibility of each residue. In the fourth category, protein folds are recognized with the combined methods of sequence alignment algorithm and threading.

The threading methods are computationally expensive because each entry of the library having thousands of possible folds is required to be aligned in all possible ways. The energy function used in the threading methods are not sophisticated enough to find the correct protein folds. When the sequence similarity is low, alignment errors can range from 3 Å to 6 Å in terms of RMSD. Protein Structure Prediction and Evaluation Computer Toolkit (PROSPECT) is one of the best threading programs in the Critical Assessment of Techniques for Protein Structure Prediction (CASP) competition (Xu et. al., 2001). PROSPECT can find the globally optimal sequence-structure alignment based on information provided by energy functions (Xu et. al., 2000). Divide-and–conquer algorithms for PROSPECT can speed up calculation since the

divide-and–conquer algorithm can discard the conformation search space which does not contain optimal alignment. Even when the sequence similarity is less than 17%, some high quality sequence-structure alignment between the target structure and template structure can be obtained.

2.4 Ab Initio Methods

Ab Initio methods can be used to predict protein structures from the sequence information when appropriate structure templates cannot be found. At first, the protein representation and the corresponding protein conformation space is defined. Then energy functions suitable for the protein conformation space are selected. Effective algorithms to minimize the energy function are determined in order to search the conformational space. The conformation minimizing the energy functions becomes one of candidate structures that are close to the native form of the target protein. The physical forces acting on the atoms of protein is the major force to determine the folding of protein sequences. All-atom based energy function models are the most effective model for protein structure prediction. Due to the complexity of all-atom based energy function models, it is computationally impossible to use this method for protein structure prediction. In order to solve this problem, most Ab Initio prediction methods restrict the conformation space to the reasonable size using reduced protein representation and select those energy functions related to the most important interactions responsible for protein folding in its native form. The ROSETTA Ab Initio method produces better results than other Ab Initio methods in the CASP4 conference (Bonneau et. al., 2001). The ROSETTA method uses the reduced representation of the protein as short segments. This representation is based on the concept that local segments have their preferences for local structure formation. The local structures corresponding to theses segments come from those found in all the known protein structure when the ROSETTA method is used (Simons et. al., 1997). The Bayesian probability of structure-sequence matches is

selected to be the energy function. This energy functions place the foundation for the Monte Carlo sampling of the reduced protein conformational space (Simons et. al., 1997). Terms favoring strands and buried hydrophobic residues are included into the non-local potential driving the protein toward native protein formation. Ab Initio methods can predict the local structure accurately with correct contacts among residuals. Prediction of interaction between distant residues generates largest sources of errors in this method.

After I explain how proteins structure can be experimentally determined, the clustering system to predict local protein structure is explained in the second part of dissertation. In the second part of dissertation, I discuss the new improved K-means clustering for sequence cluster and motif discovery. Then I explain how sequence variation for sequence clusters may influence its structural similarity. Based on the above information, the clustering system is used to carry out local protein structure prediction.

Chapter 3 Discovery of Sequence Clusters and Sequence Motifs with Improved K-means Algorithms

In this chapter, recurring sequence motifs of proteins are explored with an improved K-means clustering algorithm. Information about local protein sequence motifs is very important to the analysis of biologically significant conserved regions of protein sequences. These conserved regions can potentially determine the diverse conformation and activities of proteins. Carefully constructed sequence motifs from sequence clusters are used to predict local protein structure. The structural similarity of these recurring sequence motifs is studied in order to evaluate the correlation between sequence motifs and their structures. The evolutionary distance, which is essential for our K-means algorithm, is explained in details for the first time. A new greedy initialization method for the K-means algorithm is proposed to improve traditional K-means clustering techniques. The new initialization method tries to choose suitable initial points, which are well separated and have the potential to form high-quality clusters. Our experiments indicate that the improved K-means algorithm satisfactorily increases the percentage of sequence segments belonging to clusters with high structural similarity. Careful comparison of sequence motifs obtained by the improved and traditional algorithms also suggests that the improved K-means clustering algorithm may discover some relatively weak and subtle sequence motifs, which are undetectable by the traditional K-means algorithms. Many biochemical tests reported in the literature show that these sequence motifs are biologically meaningful. Experimental results also indicate that the improved K-means algorithm generates more detailed sequence motifs representing common structures than previous research. Furthermore, these motifs are universally conserved sequence patterns across protein families, overcoming some weak points of other popular sequence motifs. The satisfactory result of the experiment suggests that this new

K-means algorithm may be applied to other areas of bioinformatics research in order to explore the underlying relationships between data samples more effectively.

This chapter has been divided into five sections. In section 3.1, the major motif discovery methods are discussed. In section 3.2, the major characteristics of the traditional and improved K-means algorithms are compared. In section 3.3, the experimental setup is explained. In section 3.4, experimental results are presented to show that the improved K-means algorithm is better than the traditional K-means algorithm and to give evidence that our research find some previously undiscovered sequence motifs. In section 3.5, our research is compared to other state-of-art approaches in order to emphasize the advantages of our research.

3.1 Several Major Motif Discovery Methods

In this section, the advantages and disadvantages of several motif discovery methods are compared. Since clustering algorithms can provide an automatic, unsupervised discovery process for sequence motifs, the clustering algorithm is chosen as the motif discovery method in this study.

Understanding the relationship between protein structure and its sequence is one of the most important tasks of current bioinformatics research. Many biochemical tests suggest that a sequence determines conformation completely, because all the information that is necessary to specify protein interaction sites with other molecules is embedded into its amino acid sequence (Karp, 2002). This close relationship between protein sequences and structures forms the theoretical basis for exploring the sequence motifs representing a strong common structure. Various researches show that a relatively small number of structurally or functionally conserved sequence regions are available in a large number of protein families. Representation of these conserved sequence regions can range from simple sequence motifs to complex descriptors.

These descriptors are profiles, Position Specific Scoring Matrices (PSSM) (Altschul et. al, 1997) and Hidden Markov Models (HMM) (Durbin et.al, 1998). Sequence motifs and profiles obtained from biologically significant regions may be used to predict any subsequent reoccurrence of structural or functional areas on other proteins. These functional and structural areas may include enzyme-binding sites, prosthetic group attachment sites or regions involved in binding other small molecules.

PROSITE (Hulo et al., 2004), PRINTS (Attwood et al., 2002), PFAM (Sonnhammer et. al., 1998) and BLOCKS (Henikoff et al., 1999) are four popular sequence motifs. Core PROSITE sequence patterns are created from observation of short conserved sequences, which are experimentally proven significant to the biological function of certain protein families**.** Conversion of residue frequency distributions from multiple sequence alignment by a symbol comparison table produces PROSITE sequence profiles (Hulo et al., 2004). The function, binding properties and active sites of uncharacterized proteins can be revealed after comparison with PROSITE sequence patterns and profiles (Hulo et al., 2004). Analysis of three-dimensional structure of PROSITE patterns suggests that recurrent sequence motifs imply common structure and function (Hulo et al., 2004). Fingerprints from PRINTS contain several motifs from different regions of multiple sequence alignments, increasing the discriminating power to predict the existence of similar motifs because identification of individual parts of the fingerprint is mutually conditional (Attwood et al., 2002). PFAM contains HMM sequence profiles produced by multiple sequence alignment and Hidden Markov Model (Sonnhammer et. al., 1998). PFAM includes both conserved motifs and less conserved regions, which is the major difference from PROSITE and PRINTS. Since sequence motifs and profiles of PROSITE, PRINTS and PFAM are developed from multiple sequence alignments, these sequence motifs and profiles only

search conserved elements of sequence alignments from the same protein family and carry little information about conserved sequence regions, which transcend protein families. Furthermore, the knowledge about the biologically important regions or residues is the precondition of finding these motifs. As a result, the discovery of sequence motifs and profiles requires intensive human intervention.

The clustering technique is very useful for knowledge discovery, pattern recognition, data mining and image segmentation because the clustering technique is very effective to group data together with specified similar characteristics. In many applications, researchers have little prior knowledge about data and have to make as few assumptions about the data as possible (Jain, Murty and Flynn, 1999). Under these restrictions, clustering algorithms are particularly suitable to discover the underlying relationship among the data samples and assess their common characteristics. The clustering algorithm for local sequence segments aims to classify local sequence regions into groups sharing common structures or functions. Some of these groups can be defined as the sequence motifs. These sequence motifs are very useful for further analysis of functional and structural characteristics of uncharacterized protein families. These attractive characteristics allow the clustering technique to discover universally conserved and elaborate sequence motifs across protein families. While other methods to produce the popular sequence motifs require human intervention to explore the biologically significant regions of protein sequences, the clustering technique provides an automatic, unsupervised discovery process. All these advantages, in comparison to the other four methods to create popular sequence motifs, motivate us to develop an improved K-means clustering algorithm.

Han and Baker have used the K-means clustering program to find recurring local sequence motifs for proteins (Han and Baker, 1995; Han and Baker, 1996). In their work, a set of initial

points for cluster centers is chosen randomly (Han and Baker, 1995). Since the performance of K-means clustering is very sensitive to initial point selection (Jain, Murty and Flynn, 1999), their technique may not yield satisfactory results. Random selection often obtains either initial points that are close together or outliers of clusters, producing unsatisfactory partitions since initial points need to be well separated to approximate each cluster in the sparse data space. To overcome the problem of random selection, we propose the new greedy algorithm to select suitable initial points in order to allow the K-means algorithm to converge to a better local minimum (Zhong et.al, 2004a).

In our research, protein sequences are converted into sliding sequence segments. These sliding sequence segments are classified into different groups with the improved K-means clustering algorithm. The structural similarity of these groups is evaluated. The recurrent groups with high structural similarity will become the candidate to generate sequence motifs representing a common structure. Our sequence motifs are represented by the frequency profiles.

3.2 K-means Clustering Algorithms

Since the K-means clustering algorithm is chosen as the motif discovery method, first we discuss the weak points of the traditional K-means algorithms and analyze other people's efforts to explore new initialization methods. Then, we propose the improved K-means clustering algorithm for automatic motif discovery and explain its advantages.

3.2.1 Traditional K-means Clustering Algorithm

K-means clustering is computationally efficient for large data sets with both numeric and categorical attributes (Gupta, Rao and Bhatnagar, 1999). For the traditional K-means clustering algorithm, K-samples are chosen at random from the whole sample space to approximate

centroids of initial clusters. The K-means clustering algorithm then iteratively updates the centers until no reassignment of patterns to new cluster centers occurs. In every step, each sample is allocated to its closest cluster center and cluster centers are reevaluated based on current cluster memberships (Jain, Murty and Flynn, 1999). Some researchers have adopted the K-means clustering algorithm to perform knowledge discovery in bioinformatics research. Guralnik discovered a set of features that captures underlying properties of proteins, projected each protein onto these feature spaces and applied the K-means based clustering algorithm to find protein clusters (Guranlnik and Karypis, 2001). Selbig applied K-means clustering to contact environments in order to explore correlations between sequence patterns and structural motifs (Selbig and Argos, 1998).

Random, Forgy, MacQueen and Kaufman are four initialization methods for the K-means algorithm (Pena, Lozano and Larranaga, 1999). In these four initialization methods, the choice of initial data points defines deterministic mapping from the initial partition to the results since the K-means algorithm tries to find optimal local minima. Inappropriate choices of initial points in these four initialization methods may result in distorted or incorrect partitions, which are far from the globally optimal solution. A large percentage of data samples may be concentrated into small numbers of clusters while remaining clusters have a very small number of samples. Due to restriction of current protein database design and very large number of sequence segments generated by our protein dataset, it is impractical to implement Random, MacQueen and Kaufman as the initialization method for K-means clustering technique in our application. As a result, we choose Forgy as the initialization method for the traditional K-means clustering algorithm. The Forgy approach will select K samples from the database randomly as the

representation for initial cluster centers (Pena, Lozano and Larranaga, 1999). In our paper, random selection of data samples refers to the Forgy approach.

Many efforts have been taken to choose suitable initial clustering centers so that the algorithm is more likely to find the global minimum value (Jain, Murty and Flynn, 1999). Suitable initial clustering centers are far enough away to belong to different natural partitions and have the potential to create clusters with strong common characteristics. Special assumptions about the data distribution, which is the precondition for implementation of these new initialization methods, are not appropriate to our application due to complex underlying distribution patterns of our data set. Juan implemented supervised selection and the greedy interchange algorithm to improve the quality of partitioning (Juan and Vidal, 2000). In the supervised selection algorithm, a small subset of samples is marked according to the prespecified classification scheme. Then, the seeds can be chosen, class-by-class, to guarantee better dispersion than that of random selection (Juan and Vidal, 2000). This initialization method does not work well in our application, since we select 800 appropriate initial points out of 500,000 samples and the information about the underlying distribution model of samples is not available. Sun and others have created an iterative initial-points refinement algorithm to find appropriate initial sample points (Sun, Zhu and Chen, 2002). The observation that sub-sampling can give some information about the location of the data mode provides the foundation for their algorithm (Sun, Zhu and Chen, 2002). Knowledge about the true mode is critical for initialization of the general cluster algorithm. However, this method may not be suitable for our work, since our protein database contains large and complexly distributed data points. It is very difficult to discover the true modes by small sub-sampling of our data set due to the heterogeneous nature of protein data. To the best of our knowledge, our application does not satisfy the precondition to use the available

improved initialization methods. Therefore, we propose a new greedy initialization method for K-means algorithm. This new initialization method does not depend on the knowledge about the underlying distribution patterns of the data set, which is the advantage over other available improved initialization methods for the K-means algorithm.

3.2.2 New Greedy Initialization Method for the K-means Algorithm

To overcome potential problems of random initialization, the new greedy initialization method tries to choose suitable initial points so that final partitions can represent the underlying distribution of the data samples more consistently and accurately (Zhong et.al, 2004b). Each initial point is represented by one local sequence segment. In the new initialization method, the clustering algorithm will only be performed for several iterations during each run. After each run, initial points, which can be used to form the cluster with good structural similarity, become candidate points. The evolutionary distance of these candidate points is checked against that of all points already selected in the initialization array. The evolutionary distance is defined in the section 3.3.4. If the minimum evolutionary distance of new points is greater than the specified distance, these points will be added to the initialization array. Satisfaction of the minimum evolutionary distance can guarantee that each newly selected point will be well separated from all the existing points in the initialization array and will potentially belong to different natural clusters**.** This process will be repeated several times until 800 points are chosen. After this procedure, these carefully selected points can be used as the initial centers for the K-means clustering algorithm**.**

Here is an example of how this new initialization method works. Let us suppose that the structural similarity threshold is given as 65% and the distance threshold is given as 1400. After three iterations, one initial point creates the cluster with structural similarity of 67%, which is

greater than structural similarity threshold. As a result, this point will be one of possible candidates. In the second step, the evolutionary distance of this point against all the existing points in the initialization array is calculated. The minimum evolutionary distance against all the existing points is 1439, which is greater than the distance threshold. Therefore, the point is added into the initialization array. This process will continue until 800 initial points for the K-means algorithm is chosen. The pseudocode for the initialization method of the improved K-means algorithm is given in the following:

```
WHILE (the number of initial points discovered is less than the total number of clusters)
{
     Randomly select initial points whose number is equal to 800 minius the number of seeds in the
      initialization array.
     Run the traditional K-means algorithm for a fixed number of iterations on the sample
     space excluding  the clusters produced from seeds
    Assess structural similarity of clusters produced by each initial point
    IF (the structural similarity for one cluster is bigger than or equal to a given threshold)
    {
          Check the minimum distance of the point producing this cluster with existing points in
          the  initialization array
         IF (the minimum distance is bigger than threshold)
             This new point is included into the initialization array as the seed
        END IF
   }
  END IF
}
END WHILE
```

3.3 Experiment Setup

In this section, we introduce experimental parameters, the data set, and the method to generate and represent the sequence segments. Then, we discuss the cluster membership calculation for sequence segments and the structural similarity of a given cluster. Finally, we provide two measures in order to evaluate the performance of clustering algorithms.

3.3.1 Experimental Parameters

Different number of initial clusters were tried and based on these results, 800 clusters are chosen empirically. 800 clusters are relatively suitable for the K-means clustering algorithm (Jain et al., 1999) in our application based on the performance evaluation for the number of clusters with high structural similarity. Since the K-means clustering algorithm is very sensitive to starting points, the numerical stability of the cluster algorithm is estimated by performing K-means clustering five times with different random starting points. Only recurrent clusters come into the analysis of results. A structural similarity threshold is set as 70% initially. However, it took 20 days for the program to find 800 suitable initial points. To conserve the computation time, the structural similarity is set as 65% for all the experiments. Different evolutionary distances are used to evaluate their effects on clustering performance.

3.3.2 Dataset

The dataset used in this work includes 2000 protein sequences obtained from the Protein Sequence Culling Server (PISCES) (Wang and Dunbrack, 2003). This data set is the training set for local protein structure prediction, which will be introduced in the later chapter.

In this protein database, the percentage identity cutoff is 25%, the resolution cutoff is 2.2, and the R-factor cutoff is 1.0. No sequences of this database share more than 25% sequence identities. This protein database is bigger and more advanced than PDB-select 25 (Hobohm, et al., 1992) used by Han and Baker. Since PISCES uses PSI-BLAST (Altschul et al., 1997) alignments to distinguish many underlying patterns below 40% identity, PISCES produces a more rigorous non-homologous database than PDB-select 25. PISCES local alignment will not incorporate two proteins that share a common domain with sequence identity above the given

threshold (Wang and Dunbrack, 2003). This feature helps to overcome problems of PDB-REPRDB (Noguchi, Matsuda and Akiyama, 2001), which uses global alignment methods that may generate useless sequence similarities for multidomain proteins.

3.3.3 Generation and Representation of Sequence Segments

The sliding windows with ten successive residues are generated from protein sequences. Each window represents one sequence segment of ten continuous positions. Five hundred thousand sequence segments from 2290 protein sequences are produced by the sliding window method. These sequence segments of ten continuous positions are classified into different groups with the K-means algorithm.

Careful choice of representation for sequence segments can yield noticeably improved and easily understood clustering results. The frequency profile from a database of Homology-derived Secondary Structure of Proteins (HSSP) (Sander and Schneider, 1991) is constructed based on the alignment of each protein sequence from the Protein Data Bank (PDB) with all sequences considered homologous in the sequence database. In the HSSP frequency profile, the frequency for a specified amino acid residue in a given sequence position is calculated by division of the number of the specified residue by total number of residues in that position. Because the HSSP frequency profile conveys context-dependent information and the general view of conserved regions, the HSSP frequency profile is very important in exploring preferences and patterns for sequence analysis and in explaining structural roles of conserved residues. Because of many important information embedded in the HSSP profiles, the HSSP frequency profiles are chosen as the representation of sequence segments in this study.

3.3.4 Evolutionary Distance and Cluster Membership Calculation for Sequence Segments

In our K-means algorithm, a sequence segment is assigned to a specific cluster if the sequence segment is closest to the center of this specific cluster in terms of the evolutionary distance. The cluster center is represented by the centroid of all sequence segments belonging to this cluster. The shortest evolutionary distance between a sequence segment and its assigned cluster center might increase possibility for this sequence segment to share a common structure and function with other sequence segments in the same cluster. Therefore, the usage of the evolutionary distance is essential for the successful clustering of sequence segments.

The most common distance metric for continuous features is the Euclidean distance and the city block metric (Jain, Murty and Flynn, 1999). Euclidean distance can evaluate the proximity of two sequence segments in multi-dimensional feature space. However, the largest-scaled feature can dominate other features for the Euclidean distance. The city block metric is more suitable for our study since the city block metric will consider every position of the frequency profiles equally and information about the important positions is not available. Han and Baker also chose the city block metric because of complications associated with the use of Euclidean metric for clustering algorithms (Han and Baker, 1995). The equation 1 defines the evolutionary distance between two sequence segments.

$$\text{Evolutionary distance} = \sum_{i=1}^{L} \sum_{j=1}^{N} \left| F_k(i, j) - F_M(i, j) \right| \quad (1)$$

Where L is the size of window and N is equal to 20. $F_k(i, j)$ is the value of matrix representing the first sequence segment at row i and column j. $F_m(i, j)$ is the value of matrix representing the second sequence segment at row i and column j. The evoluationary distance already satifies four conditions for a mathematical metrics.

Expecially, the evoluationary distance obeys the famous triangle inequality. Triangle inequality has been proved by Salas, Hille and Etgen (2003).

3.3.5 Secondary Structure Assignment

DSSP (Kabsh and Sander, 1983), DEFINE (Richards and Kundrot, 1988) and STRIDE (Frishman and Args, 1995) are methods used to determine the secondary structure from the experimentally defined tertiary structure. The DSSP initially assigns the secondary structure to eight different classes. Before going through the clustering process, the structure is converted to three classes based on the following method: H, G and I to H; B and E to E; all others to C. In this paper, H represents helices; E represents sheets and C represents coils.

3.3.6 Measure of Structural Similarity for a Given Cluster

The formula 2 calculates the level of structural similarity (Han and Baker, 1996; Henikoff et al., 1999):

$$\text{Structural similary for a given cluster (\%)} = \frac{\sum_{i=1}^{ws} \max(P_{i,H}, P_{i,E}, P_{i,C})}{ws} \quad (2)$$

ws is the window size. $P_{(i,H)}$ is the frequency of occurrence of helices among the sequence segments for the cluster in position i. $P_{(i,E)}$ is the frequency of occurrence of sheets among the sequence segments for the cluster in position i. $P_{(i,C)}$ is the frequency of occurrence of coils among the sequence segments for the cluster in position i. The secondary structure with the maximum frequency is used for representing the common structure in that position. For example, $P_{(5,H)}$ = 80%, $P_{(5,E)}$ = 15% and $P_{(5,C)}$ = 5%. $P_{(5,H)}$ = 80% represents that the frequency of occurrence of helices among the sequence segments for the clusters is 80% in position 5 of the window. As a result, max($P_{(5,H)}$, $P_{(5,E)}$, $P_{(5,C)}$) is 80% with the representative structure as helices.

The average results of the max frequency from all positions of a given window show the structural similarity level for a given cluster. If the structural similarity for secondary structure within the cluster exceeds 70%, the cluster can be considered structurally identical or similar (Sander and Schneifer, 1991). If the structural similarity for secondary structure within the cluster is between 60% and 70%, the cluster can be considered weakly structurally similar.

3.3.7 Evaluation of Performance for the Clustering Algorithm and Generation of Frequency Profiles for Sequence Motifs

The percentage of sequence segments belonging to clusters with high structural similarity and the number of clusters with high structural similarity are two measures to evaluate the performance for the clustering algorithm. In the section of experimental results, the percentage of sequence segments belonging to clusters with high structural similarity and the number of clusters with high structural similarity are averaged from five-times running results**.** Improved average percentage of sequence segments belonging to clusters with high structural similarity indicates that the clustering algorithm can increase its effectiveness to classify data with specified similar characteristics. If new sequence patterns are discovered from the increased number of clusters with high structural similarity, the clustering algorithm can reveal more underlying relationships between data samples. The percentage of sequence segments belonging to clusters with the structural similarity greater than 60% is calculated by division of the sum of all sequence segments belonging to clusters with the structural similarity greater than 60% by total number of sequence segments in the database. During the process of generating frequency profiles for sequence motifs, the frequency for the specified amino acid residue in a given window position for a cluster is calculated by division of the number of specified residues by

total number of residues in that position. Only recurrent clusters with the structural similarity over 60% from five runs are considered good enough to generate sequence motifs.

3.4. Experimental Results

In this section, we compare the experimental results of the traditional and improved K-means algorithm. We also discuss the sequence motifs generated by the improved K-means algorithm and use the biochemical experiment to support biological meanings of out sequence motifs.

3.4.1 Comparison of Performance for the Traditional and Improved K-means Algorithm

In Table 1, the average percentage of sequence segments belonging to clusters with high structural similarity for the traditional and improved K-means algorithm is given.

Table 1. Comparison of the Percentage of Sequence Segments Belonging to Clusters with High Structural Similarity

Different Algorithms	>60%	>60%	>70%	>70%
New 1100	28.57%	1.13	11.67%	0.74
Traditional	30.35%	0.98	14.16%	0.68
New 1200	31.78%	0.62	12.88%	0.45
New 1300	32.37%	0.70	13.99%	0.48
New 1400	34.33%	0.54	15.10%	0.37
New 1500	35.86%	0.56	15.67%	0.42

The first column of Table 1 shows the algorithm with different parameters. “Traditional” refers to the traditional K-means algorithm, which randomly selects the initial points from the whole sample space. “New 1100” illustrates the improved K-means algorithm choosing initial points, which can potentially form clusters with good structural similarity. The minimum evolutionary distances among these points for the initialization array are at least 1100. “New 1200,” “New 1300,” “New 1400,” and “New 1500” share the similar idea with “New 1100.” The only difference from “New 1100” is the minimum evolutionary distance among initial points in

the initialization array. “New 1300” has the minimum evolutionary distance of at least 1300. “New 1400” has the minimum evolutionary distance of at least 1400. “New 1500” has the minimum evolutionary distance of at least 1500. The second column of Table 1 gives the average percentage of sequence segments belonging to clusters with the structural similarity greater than 60% from five runs. The third column of Table 1 gives the standard deviation of the percentage of sequence segments belonging to clusters with structural similarity greater than 60%. The fourth column of Table 1 gives the average percentage of sequence segments belonging to clusters with structural similarity greater than 70% from five runs. The fifth column of Table 1 gives the standard deviation of the percentage of sequence segments belonging to clusters with the structural similarity greater than 70%.

Our experimental results show an average of 40 clusters out of 800 clusters is empty after the first iteration of the traditional K-means algorithm with random selection of initial points (Zhong et.al, 2004a). Further analysis indicates that most initial points that create these 40 clusters come from outliers of clusters. Outliers of clusters refer to sequence segments, which are far away from centers of natural clusters. Analysis of the clustering process of the traditional clustering algorithm also reveals that some of the initial points are very close to each other, creating strong interferences with each other. Strong interferences among initial points will affect final partitioning negatively. The results of Table 1 show the average percentage of sequence segments belonging to clusters with structural similarity greater than 60% steadily improves with increasing minimum evolutionary distances among initial points. This improved percentage results from decreased interferences among initial points when the evolutionary distances among initial points are increased. The average percentage performance of “New 1100” is worse than that of the traditional K-means algorithm as a result of strong interferences among initial points,

which are too close to each other. “New 1500” increases the average percentage of sequence segments belonging to clusters with the structural similarity greater than 60% by almost 5.5% and improves the average percentage of sequence segments belonging to clusters with the structural similarity greater than 70% by 1.5%. Furthermore, “New 1500” reduces the standard deviation for the percentage of sequence segments belonging to clusters with the structural similarity greater than 60%. The increased average percentage and decreased standard deviation suggest that the improved K-means algorithm performs better and more consistently than the traditional algorithm because the improved K-means algorithm avoids outliers of clusters and keeps initial points as far as possible. Table 2 shows the number of clusters exceeding given structural similarity thresholds for the traditional and improved K-means algorithm.

Table 2. Comparison of the Number of Clusters with High Structural Similarity

Different Algorithms	>60%	>60%	>70%	>70%
New 1100	224	3.93	83	2.56
Traditional	211	4.15	80	2.39
New 1200	235	3.46	82	2.28
New 1300	242	3.32	85	2.25
New 1400	246	2.98	88	2.13
New 1500	253	3.01	92	2.06

The first column of Table 2 is the same as that of Table 1. The second column of Table 2 shows the average number of clusters with the structural similarity greater than 60% from five runs. The third column of Table 2 shows the standard deviation for the number of clusters with the structural similarity greater than 60%. The fourth column of Table 2 shows the average number of clusters with the structural similarity greater than 70% from five runs. The fifth column of Table 2 indicates the standard deviation for the number of clusters with structural similarity greater than 70%. “New 1500” increases the average number of clusters with structural

similarity greater than 60% by 42. Comparison between sequence motifs obtained by both algorithms suggests that the improved K-means clustering algorithm may discover some relatively weak and subtle sequence motifs. These motifs are undetectable by the traditional K-means algorithm because random selection of points may choose two starting points, which are within one natural cluster. For example, some of the weak amphipathic helices and sheets are not discovered by the traditional K-means algorithms. In addition, the number of repeated substitution patterns of sequence motifs found by the traditional K-means algorithms is less than that of the improved K-means algorithms.

3.4.2 Sequence Motifs

The following format is used for representation of each sequence motif table:

The average number of sequence segments used to generate the given motif and their average structural similarity are indicated above the columns of each motif table.

- The first column of each motif table shows the position of amino acid profiles in each local sequence motif with ten consecutive positions.
- The second column of each motif table shows the types of amino acids in the given position. The amino acid appearing with the frequency greater than 0.1 are indicated by the upper case. The amino acid with the upper case emphasizes its high occurrence rate in that position. The amino acids appearing with the frequency between 0.08 and 0.1 are indicated by the lower case.
- The third column shows the variability. Variability indicates the number of amino acids occurring with the frequency greater than 0.05.

- The fourth column indicates the hydrophobicity index. The hydrophobicity index is the sum of the frequencies of occurrence of alanine, valine, isoleucine, leucine, methionine, proline, phenylalanine, and tryptophan.
- The fifth column indicates the representative secondary structure in that position.

Motif table for Pattern 1
Helices with conserved L

Number of segments: 1086 Structural homology: 61.1%				
P	Patterns	V	H	S
1	v	12	0.38	H
2	aE	11	0.33	H
3	l	12	0.33	H
4	Lv	8	0.45	H
5	ael	11	0.35	H
6	AL	9	0.37	H
7	L	1	0.92	H
8	L	1	0.89	H
9	adE	10	0.29	H
10	a	10	0.31	H

Motif table for Pattern 2
Coil with low hydrophobicity

Number of segments: 222 Structural homology: 67.0%				
P	Patterns	V	H	S
1	rNqgs	11	0.13	C
2	G	2	0.11	C
3	NDEk	9	0.18	C
4	egps	9	0.29	C
5	ag	9	0.38	C
6	ark	11	0.29	C
7	aNdes	8	0.22	C
8	agPs	8	0.28	C
9	Ekv	10	0.32	C
10	agsT	8	0.25	C

Motif table for Pattern 3
Coil with conserved N

Number of segments: 242 Structural homology: 66.0%				
P	Patterns	V	H	S
1	NgsTY	7	0.21	C
2	NStY	6	0.2	C
3	NdgSTY	7	0.21	C
4	NgSTy	7	0.2	C
5	NgSTy	7	0.2	C
6	NdgsT	7	0.22	C
7	NgSty	7	0.2	C
8	aNdgsy	9	0.24	C
9	NdgstY	7	0.18	C
10	Ngy	8	0.21	C

More than 190 local sequence motifs indicating common structure are discovered in this study. These 190 sequence motifs have been grouped into 27 major patterns according to their common characteristics. One representative of each group is chosen to show the sequence pattern of this group. However, there is a lot of ambiguity in these patterns like words in a dictionary that have multiple meanings. Since the statistics of the structural database indicate the average length of helices is 10, 70% of the sequence motifs generated by the K-means clustering algorithm with the window size of 10 are related to helices. Analysis of related biochemical studies indicates that patterns obtained by the K-means algorithm may play vital roles in intramolecular interactions, which decide the structure and function of proteins. These patterns also influence intermolecular interaction, which affect how proteins communicate with other molecules. Furthermore, analysis of these sequence motifs provides important insight into the

degrees to which changes in the primary sequence are tolerated. This knowledge can help us understand structurally conservative substitutions of 20 amino acids during the evolutionary process.

Pattern 4, 5 and 6 contain conserved glutamic acid, lysine or serine. These three amino acids are polarly charged residues with relatively strong organic acids and bases. As a result, these amino acids can establish ionic bonds with other charged molecules in the cells and play important roles in catalysis and salt bridges (Berg, Tymoczko and Stryer, 2002) These charged amino acids are also important to decide the characteristics of protein surfaces, which act as the major functional locations for many proteins (Robertson, 2002)

Motif table for Pattern 4
Polar helices with conserved E and K

Number of segments: 436 Structural homology: 63.0%				
P	Patterns	V	H	S
1	AREK	9	0.26	H
2	ArqEK	7	0.24	H
3	AREK	7	0.29	H
4	Aelkv	8	0.41	H
5	ArEK	8	0.25	H
6	AqEK	7	0.24	H
7	Aelk	8	0.41	H
8	arEK	8	0.26	H
9	ArEK	7	0.23	H
10	REK	9	0.22	H

Motif table for Pattern 5
Amphipathic helices with conserved E and K

Number of segments: 778 Structural homology: 77.8%				
P	Patterns	V	H	S
1	ADEk	8	0.18	H
2	arEl	9	0.32	H
3	ILfv	4	0.80	H
4	ADqEK	6	0.24	H
5	ArEK	7	0.28	H
6	IL	5	0.84	H
7	AREK	7	0.32	H
8	AdqEK	7	0.20	H
9	AEK	7	0.29	H
10	alv	10	0.45	H

Motif table for Pattern 6
Coil-sheet with conserved E and S

Number of segments: 447 Structural homology: 66.6%				
P	Patterns	V	H	S
1	neS	9	0.25	C
2	aneGs	9	0.21	C
3	egS	10	0.23	C
4	aDegS	8	0.21	C
5	Ks	10	0.17	C
6	eSt	9	0.24	E
7	ILV	5	0.69	E
8	ReKT	8	0.27	E
9	IlV	3	0.89	E
10	esT	6	0.28	E

The hydrophobic property of amino acid side chains can affect protein conformation and function (Kyte and Doolitle, 1982; Zimmerman, Eliezer and Simha, 1968). Thermodynamics show that polar or hydrophilic residues are placed onto the surface of protein interacting with surrounding water and nonpolar residues tend to gather within the interior of most soluble proteins, connecting with one another as the result of van der Waals forces and hydrophobic interactions (Berg, 2002; Kauzmann, 1959; Privalov, 1997). These hydrophobic interactions

among nonpolar residues increase the overall stability of the protein. For many enzymes, reactive polar residues can move into the nonpolar interior in order to increase chemical reaction between polar groups (Karp, 2002). Since the level of hydrophobicity plays important roles in determining the structure and activities of proteins, special attentions have been paid to analyze the hydrophobic and hydrophilic patterns of the sequence motifs. Many patterns related to helices show pronounced amphipathicity such as Pattern 7, 8 and 9 since amphipathic helices are one of the common structural motifs in proteins (Segrest, Loof, and Dohlman, 1990). In the soluble protein, the hydrophobic face of helices is buried into the protein interior and the polar face can project into its polar surrounding (Berg, Tymoczko and Stryer, 2002). Pattern 7, 8 and 9 show that hydrophobic amino acids are regularly arranged three or four positions apart. Amphipathic helices are first found in myoglobin. Several methods are proposed to identify these amphipathic helices (Finer-Moore and Stroud, 1984; Schiffer and Edmundson, 1967). Possible functions of these amphipathic helices have been experimentally tested (DeGrado, 1988; Kaiser and Kezdy, 1984). Peptides that show amphipathic structural motifs have been widely adopted as the model system to understand problems associated with protein folding and stability (Chen, Mant and Hodges, 2002; Mant, Zhou and Hodges, 1993)

Motif table for Pattern 7
Amphipathic helics with conserved A and L

Number of segments: 995 Structural homology: 70.2%				
P	Patterns	V	H	S
1	aL	8	0.46	H
2	Ae	12	0.38	H
3	ALv	9	0.53	H
4	A	1	0.85	H
5	Arel	10	0.38	H
6	AreL	8	0.41	H
7	L	1	0.9	H
8	A	9	0.4	H
9	AEk	8	0.31	H
10	a	11	0.36	H

Motif table for Pattern 8
Amphipathic helices with repeating ILV and DEK substitution patterns

Number of segments: 693 Structural homology: 74.4%				
P	Patterns	V	H	S
1	ILV	4	0.73	H
2	arDEk	8	0.2	H
3	Arl	8	0.4	H
4	ILV	5	0.8	H
5	RK	6	0.21	H
6	ArdEK	8	0.26	H
7	ILV	6	0.78	H
8	rl	8	0.4	H
9	adEK	9	0.19	H
10	adEk	9	0.28	H

Motif table for Pattern 9
Helices with very conserved A

Number of segments: 1356 Structural homology: 74.0%				
P	Patterns	V	H	S
1	Ae	11	0.35	H
2	Ad	8	0.39	H
3	AiLV	8	0.52	H
4	Al	11	0.4	H
5	Ael	11	0.42	H
6	A	1	0.82	H
7	AiLV	6	0.58	H
8	Arek	8	0.39	H
9	AL	7	0.49	H
10	A	1	0.83	H

Pattern 11 shows helices with very high hydrophobicity. This may suggest that Pattern 11 may be located at the core of proteins, linking its NH and CO groups with hydrogen bonding. Pattern 10 reveals amphiphilic helices with very low hydrophobicity. Pattern 10 may point to polar solution. Amphiphilic helices may determine the functions of representative apolipoproteins, peptide toxins and peptide hormones. By increasing the amphipilicity of the structurally significant regions of the molecule, the biological activity of the peptide can surpass naturally occurring polypeptide (Kaiser and Kézdy, 1983) As a result, amphiphilic helices are very important for protein design projects (DeGrado, 1988).

Motif table for Pattern 10
Helices with very low hydrophobicity

Number of segments: 583 Structural homology: 63.3%				
P	Patterns	V	H	S
1	elk	7	0.44	H
2	EK	8	0.22	H
3	aElk	9	0.35	H
4	ILv	8	0.58	H
5	qEk	9	0.27	H
6	AEK	7	0.28	H
7	E	1	0.04	H
8	aIkV	8	0.54	H
9	aEks	7	0.27	H
10	AReK	8	0.3	H

Motif table for Pattern 11
Helices with high hydrophobicity

Number of segments: 620 Structural homology: 88.5%				
P	Patterns	V	H	S
1	AiLfv	7	0.66	H
2	AILV	7	0.72	H
3	AiLV	8	0.67	H
4	AILV	7	0.68	H
5	AILV	6	0.74	H
6	AgILV	8	0.71	H
7	AILfV	7	0.71	H
8	AgILv	8	0.66	H
9	aILFV	6	0.74	H
10	AILfv	6	0.71	H

Many patterns associated with coils show very low hydrophobicity. Coils are located on the surfaces of proteins and are sometimes involved in chemical interaction between proteins and other molecules (Berg, Tymoczko and Stryer, 2002; Hulo et al., 2004).Many patterns associated with sheets have high levels of hydrophobicity since hydrophobic amino acids are statistically preferred for the sheet structure (Hutchinson and Thornton, 1994; Lifson and Sander, 1997).

Motif table for Pattern 12
Coil with conserved S and T

Number of segments: 291 Structural homology: 63.6%				
P	Patterns	V	H	S
1	aNsT	7	0.29	C
2	nSt	9	0.25	C
3	St	4	0.17	C
4	anST	7	0.26	C
5	gST	7	0.25	C
6	anST	8	0.25	C
7	psT	9	0.28	C
8	aST	7	0.24	C
9	aST	7	0.2	C
10	ST	8	0.28	C

Motif table for Pattern 13
Amphipathic sheet

Number of segments: 467 Structural homology: 70.0%				
P	Patterns	V	H	S
1	NDg	10	0.19	C
2	Agkv	10	0.36	C
3	RK	6	0.18	E
4	ILV	4	0.86	E
5	ILV	5	0.76	E
6	AiLV	6	0.62	E
7	IlV	5	0.74	E
8	akSt	8	0.25	E
9	Deps	8	0.27	C
10	nDgs	8	0.26	C

Pattern 14 shows interesting alternating hydrophobic-polar residues. Pattern 15 and 16 indicate the sheet-coil with clear hydrophobicity transition. Transitional patterns for hydrophobicity found in our sequence motifs are reasonable because hydrophobic amino acids are preferred for sheets and hydrophilic amino acids frequently occur in coils.

Motif table for Pattern 14
Sheet with alternating hydrophobic-polar from position 1 to position 6

Number of segments: 475 Structural homology: 65.3%				
P	Patterns	V	H	S
1	ILfV	4	0.8	E
2	rESTv	7	0.3	E
3	ILFV	6	0.79	E
4	DET	8	0.23	E
5	ILV	6	0.64	E
6	DES	11	0.17	C
7	anDEGp	8	0.16	C
8	DEG	7	0.18	C
9	dGk	10	0.24	C
10	iLv	8	0.52	E

Motif table for Pattern 15
Sheet-coil with clear hydrophocity transition

Number of segments: 480 Structural homology: 67.1%				
P	Patterns	V	H	S
1	ads	11	0.26	C
2	Av	9	0.41	E
3	ILV	5	0.74	E
4	ILV	4	0.87	E
5	ILV	4	0.8	E
6	Ast	6	0.41	E
7	aNDe	9	0.21	C
8	DEgs	10	0.21	C
9	DEgs	10	0.23	C
10	Dgps	10	0.24	C

Motif table for Pattern 16
Sheet-coil with clear hydrophobicity transition

Number of segments: 536 Structural homology: 68.3%				
P	Patterns	V	H	S
1	adV	9	0.42	E
2	ILV	5	0.83	E
3	ILV	4	0.83	E
4	ILV	4	0.83	E
5	AsT	5	0.35	E
6	Adest	10	0.23	C
7	ADegs	9	0.27	C
8	adeGs	8	0.25	C
9	Deps	9	0.25	C
10	degp	10	0.32	C

Pattern 17 illustrates the coils containing conserved glycines in several positions. Many other patterns also contain conserved glycine residues. The side chain of glycine only has one hydrogen atom. The properties of lacking side chains allow the protein backbone to move and approach other backbones very closely (Karp, 2002). As a result, it is worthwhile to study the position of conserved glycine in the sequence patterns. Pattern 18 and 19 contain conserved proline residues in several positions. Proline does not easily fit into an ordered secondary structure because its ring structure increases the restriction on its conformation (Berg, Tymoczko and Stryer, 2002). As a result, the frequency of proline is low for patterns related with helices and sheets and is high for patterns related with coils.

Motif table for Pattern 17
Coil with conserved G

Number of segments: 276 Structural homology: 70.2%				
P	Patterns	V	H	S
1	aqGs	8	0.25	C
2	AGs	8	0.27	C
3	aG	7	0.32	C
4	aG	8	0.24	C
5	aqGt	7	0.28	C
6	aGp	8	0.24	C
7	GSt	8	0.17	C
8	GS	9	0.23	C
9	aqG	8	0.25	C
10	dGp	9	0.23	C

Motif table for Pattern 18
Coil with conserved P

Number of segments: 439 Structural homology: 69.7%				
P	Patterns	V	H	S
1		11	0.36	C
2	il	12	0.42	C
3	p	10	0.36	C
4	P	1	0.09	C
5	p	10	0.36	C
6	p	11	0.35	C
7	v	9	0.42	C
8	P	1	0.1	C
9	ly	10	0.36	C
10	Ls	11	0.4	C

Motif table for Pattern 19
Coil with conserved E and P

Number of segments: 238 Structural homology: 72.8%				
P	Patterns	V	H	S
1	EGk	10	0.21	C
2	kps	10	0.31	C
3	EP	10	0.29	C
4	EP	10	0.23	C
5	ILPv	7	0.49	C
6	P	1	0.08	C
7	AdEp	10	0.3	C
8	AdePs	8	0.31	C
9	EKs	10	0.27	C
10	AEk	10	0.23	C

Pattern 20 and 21 give helices-coils motifs. Transitional regions between helices and coils contain conserved glycine since glycine favors disruption of the helices. Helix-termination rules of thumb show helix termination by glycine and proline is anticipated (Aurora and Rose, 1998). Many patterns also show very similar substitution patterns at several positions such as Patterns 22, 23 and 26. These similar substitution patterns can provide insights into conserved substitution patterns, which can preserve the structure of proteins.

Motif table for Pattern 20
Helices-coil

Number of segments: 1661 Structural homology: 70.4%				
P	Patterns	V	H	S
1	ardEK	8	0.26	H
2	AL	8	0.46	H
3	L	4	0.85	H
4	AREK	9	0.3	H
5	AEK	6	0.27	H
6	AL	7	0.41	C
7	G	1	0.09	C
8	ILV	6	0.63	C
9	dEt	11	0.27	C
10	ilV	7	0.5	E

Motif table for Pattern 21
Helices-coil-sheet with conserved L

Number of segments: 1486 Structural homology: 67.5%				
P	Patterns	V	H	S
1	AeL	8	0.45	H
2	L	4	0.85	H
3	AREK	7	0.28	H
4	AEK	7	0.27	H
5	AL	9	0.42	C
6	G	1	0.08	C
7	ILfV	6	0.67	C
8	dEkt	10	0.26	C
9	iV	8	0.51	E
10	iV	9	0.46	E

Motif table for Pattern 22
Helices with repeated AST substitution patterns

Number of segments: 415 Structural homology: 68.8%				
P	Patterns	V	H	S
1	AStv	4	0.51	H
2	AST	5	0.5	H
3	AiLv	8	0.55	H
4	AgS	5	0.55	H
5	AgSt	5	0.52	H
6	aILV	5	0.73	H
7	aLst	9	0.38	H
8	As	6	0.54	H
9	aLv	6	0.78	H
10	as	11	0.3	H

Motif table for Pattern 23
Sheet with repeating ILV

Number of segments: 568 Structural homology: 67.8%				
P	Patterns	V	H	S
1	ILV	3	0.8	E
2	Ekt	8	0.24	E
3	ilV	8	0.48	E
4	nDEG	8	0.14	C
5	NDG	7	0.13	C
6	deGK	8	0.18	C
7	eKt	10	0.23	E
8	ILfV	6	0.74	E
9	Tv	8	0.32	E
10	ILfV	5	0.78	E

Motif table for Pattern 24
Coil-helices

Number of segments: 908 Structural homology: 77.1%				
P	Patterns	V	H	S
1	ae	10	0.35	C
2		11	0.29	C
3	ILv	5	0.84	C
4	DPST	6	0.1	C
5	aDEkP	6	0.19	H
6	aDE	5	0.14	H
7	DQE	7	0.29	H
8	aILkV	7	0.63	H
9	ArdqEK	7	0.24	H
10	AREK	7	0.3	H

Motif table for Pattern 25
Coil-sheet-coil

Number of segments: 472 Structural homology: 63.3%				
P	Patterns	V	H	S
1	Adg	8	0.37	C
2	REK	9	0.24	E
3	IV	3	0.88	E
4	ILV	7	0.59	E
5	Arek	9	0.32	E
6	ILV	3	0.79	E
7	NDEs	6	0.16	E
8	dL	11	0.39	C
9	anDek	9	0.26	C
10	DE	9	0.18	C

Motif table for Pattern 26
Coil-sheet-coil with conserved ILV

Number of segments: 784 Structural homology: 65.3%				
P	Patterns	V	H	S
1	rdegks	10	0.22	C
2	DeGK	8	0.17	C
3	aGkpv	9	0.3	C
4	RK	7	0.21	E
5	ILV	5	0.79	E
6	ILV	6	0.69	E
7	aILV	6	0.77	E
8	ILV	4	0.8	E
9	AST	8	0.29	E
10	gs	11	0.28	C

Motif table for Pattern 27
Coil-sheet with repeating ILV

Number of segments: 535 Structural homology: 71.8%				
P	Patterns	V	H	S
1	adgs	9	0.3	C
2	dEgKP	9	0.18	C
3	NDGK	8	0.15	C
4	AilV	8	0.49	C
5	RK	8	0.24	E
6	ILV	5	0.84	E
7	ILV	3	0.84	E
8	AiLV	6	0.59	E
9	ILV	6	0.73	E
10	AndeST	8	0.27	E

3.5 Result Comparison with Other Research

In this section, we compare our work with other state-of-the-art approaches. Our results reveal much more detailed hydrophobicity patterns for helices, sheets and coils than the previous study (Han and Baker, 1995). These elaborate hydrophobicity patterns are supported by various biochemical experiments. Increased information about hydrophobicity patterns associated with these sequence motifs can expand our knowledge of how proteins fold and how proteins interact

with each other. Furthermore, the analysis of discovered sequence motifs shows that some elaborate and subtle sequence patterns such as Pattern 1, 9, 22 have never been reported in previous works. Especially, increased number of repeated substitution patterns reported in this study may provide additionally strong evidences for structurally conservative substitutions during the evolutionary process for protein families.

The sequence motifs discovered in this study indicate conserved residues that are structurally and functionally important across protein families because protein sequences used in this study share less than 25% sequence identities. These important features from our sequence motifs may help to compensate for some of the weak points of those created by PROSITE, PRINTS, PFAM and BLOCKS (Attwood et al., 2002; Henikoff, Henikoff and Pietrokovski, 1999; Sonnhammer et.al., 1998). Our sequence motifs may reflect general structural or functional characteristics shared by different protein families while sequence motifs from PROSITE, PRINTS, PFAM and BLOCKS represent structural or functional constraints specific to a particular protein family. Due to the high throughput sequencing techniques, the number of known protein sequences has increased rapidly in recent years. However, information about functionally significant regions of these new proteins may not be available. As a result, automatic discovery of biologically important sequence motifs in this study is a much more powerful tool to explore underlying correlations between protein sequences, structures and functions than other methods requiring existing human knowledge.

In this study, the new initialization method for the K-means algorithm has been proposed to solve problems associated with random selection. In the new initialization method, we try to choose suitable initial points, which are well separated and have the potential to form a high-quality cluster. Many biochemical tests indicate that discovered sequence motifs are biologically

meaningful. Analysis of sequence motifs also shows the improved K-means algorithm may detect some very subtle sequence motifs overlooked by the traditional algorithm. `The reasonable experimental results show the improved K-means clustering technique is effective in classifying data with specified similar biological characteristics and in discovering the underlying relationship among the data samples. The discovered sequence motifs across protein families may overcome the shortcomings of other popular sequence motifs. Because the dataset from PISCES has several advantages over other existing databases, sequence motifs discovered in this process can reveal more patterns that are meaningful during the process of evolution than other studies. Since the K-means algorithm is a very powerful tool for data mining problems, the improved K-means algorithm may be useful for other important bioinformatics applications.

Chapter 4 Parallel K-means Algorithm using Pthread and OpenMP over Hyper-Threading Technology

In our study, the cluster number of 800 is chosen empirically. However, 800 may not be the optimal cluster number. Therefore, the improved K-means algorithm will be run several times with different values of k in order to discover the most suitable number of clusters. With the information about the optimal cluster number, clustering results may be potentially closest to underlying distribution patterns of the sample space. However, the time spent searching for the good initial points grows substantially when the minimum evolutionary distance and structural homology threshold are increased. For example, it will take 18 days to obtain appropriate initial points with the distance threshold of 1500 in the very large sample space. Due to the time and processing power constraints, the search for the optimal cluster number has not been completed. The long searching time for initial points motivates us to implement the parallel K-means algorithm in order to reduce the searching time for suitable initial points to one to two days. The parallelization of the improved K-means algorithm will make exploration of the optimal cluster number possible. We predict that the performance gains for the improved K-means algorithm will be increased further after the optimal cluster number is found. As a result, Pthread and OpenMP are employed to parallelize K-means clustering algorithm in the Hyper-Threading enabled Intel architecture. Speedup for 16 Pthreads is 4.3 and speedup for 16 OpenMP threads is 4 in the 4 processors shared memory architecture. With the new parallel K-means algorithm, K-means clustering can be performed for multiple times in reasonable amount of time. Our research also shows that Hyper-Threading technology for Intel architecture is efficient for parallel biological algorithms.

In this chapter, two important parallelization techniques for the K-means clustering

algorithm are discussed. Then programming environment and implementation details are explained. Finally, experimental results for speedup values are presented.

4.1 Parallelization

Testing the K-means clustering algorithm for sequence segments is a very slow and time consuming task because a large data set of thousands of amino acids and different algorithms have to be attempted for many times. However, the natural characteristics of the K-means algorithm allow itself to be easily parallelized because of its inherent data parallelism properties. Once parallelism is incorporated into the K-means algorithm, significant amounts of training time can be saved.

Data partitioning and task partitioning are two important parallelization techniques for the K-means clustering algorithm. In data partitioning parallelism, each processor with the same copy of clusters' centroid and frequency profiles works on one portion of the training data. After one iteration, the results from all processors are accumulated to update clusters' centroid and frequency profiles. In task partitioning parallelism, tasks are partitioned among the processors based on the architecture of k-means algorithm. Since the number for clustering is small and the number of processors is fixed in our approach, data-partitioning parallelism is more suitable to our application.

4.2 Hyper-Threading Technology

The Simultaneous Multithreading (SMT) is a method that allows multiple threads to issue instructions in each cycle. SMT maximizes performance and power consumption of the CPU. It has been identified as one of the best parallel multithreading techniques among the thread level parallelism techniques (Eggers et al., 1997). Hyper-Threading had developed the SMT for the

Intel architecture on the Intel(R) Xeon™ (Marr et al., 2002). In the Hyper-Threading enabled architecture, a single processor can be divided into multiple logical processors when needed. These logical processors can execute the instructions simultaneously. While each logical processor shares the physical execution resources efficiently, it keeps its own copy of the architecture state. Therefore, Hyper-Threading gives two virtual processors out of one physical processor. Each logical processor performs at approximately 60-70% of the capacity of one physical processor (Marr et al., 2002). Two physical processors with Hyper-Threading technology are shown in Figure 1. Programs must be parallelized and be executed in multiple threads in order to obtain the performance gains that Hyper-Threading Technology brings. Hyper-Threading Technology can be applied both data partitioning parallelism and task partitioning.

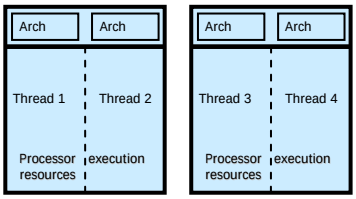

Figure 1. Two Physical Processors and Four Logical Processors

4.3 Pthread and OpenMP

Multiple threads bring parallelism for sequential programs. Thread usage is based on shared memory. Two popular parallelization methods used on shared memory are POSIX threads (Pthreads) and OpenMP. Pthreads are a very popular API for threading an application (Butenhof, 1997). OpenMP API is a multi-platform shared-memory parallel programming, which supports C/C++ and FORTRAN (Chandra et al., 2000).

Parallelizing a sequential program with OpenMP is much easier than that with Pthread because when Pthreads are used, the programmer has to deal with low-level details of thread creation, management and synchronization. Even though OpenMP is generally more suitable for data parallelization, this principle may not be applied to some applications. Therefore, we still want to compare the performance of Pthread and OpenMP in this study.

In our project, these two different parallelization methods are used separately on the same K-means clustering algorithm and the performance for two parallelization methods are compared (Zhong et.al, Accepted for Publication). Hyper-Threading Technology enabled architecture is the test bed for both methods. The performance results are very good when Hyper-Threading is used.

Other researchers have used OpenMP and MPI for paralleling neural networks. Johansson and Lansner (Johansson and Lansner, 2001) have implemented a parallel Bayesian Neural Network with Hypercolumns using OpenMP and MPI. It is shown that OpenMP is a good alternative for a medium sized Bayesian Confidence Propagation Neural Network (BCPNN) while MPI is an alternative for a large number of processors (Johansson and Lansner, 2001). The problem size has to increase substantially when the number of processors goes up in order to keep linear speed up when MPI is used (Thulasiram, Rahman and Thulasiraman, 2003).

4.4 Programming Environment and Implementation Details

An Intel® OpenMP C++/Fortran compiler for Hyper-Threading technology is used to test Pthreads and OpenMP performance in our experiments. This compiler has advanced optimization techniques for the Intel processor (Tian et al., 2002). Speedup and program execution time for Pthreads and OpenMP are measured.

The “poweredge6600 server” with four processors from Dell® is used in this study. Because of the Hyper-Threading technology, it behaves like eight logical processors. Eight or more

threads are used as shown in Figure 2. The server architecture is optimized for four Intel Xeon processor symmetric multi-processing (SMP). The operating system is Linux.

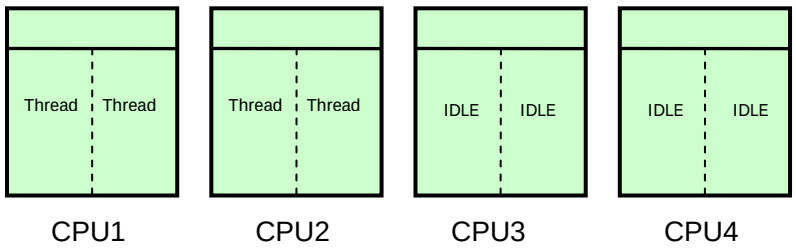

Figure 2. Four Physical Processors Behaving Like Eight Logical Processors

In order to show how our parallel algorithm works, an example for five threads is illustrated in Figure 3 when Pthread is used (Zhong et.al, 2004c). One of the threads is called the master thread and the others are referred to as slave threads. The parallelization algorithm is explained gradually below. The whole data file has been divided into several sections.

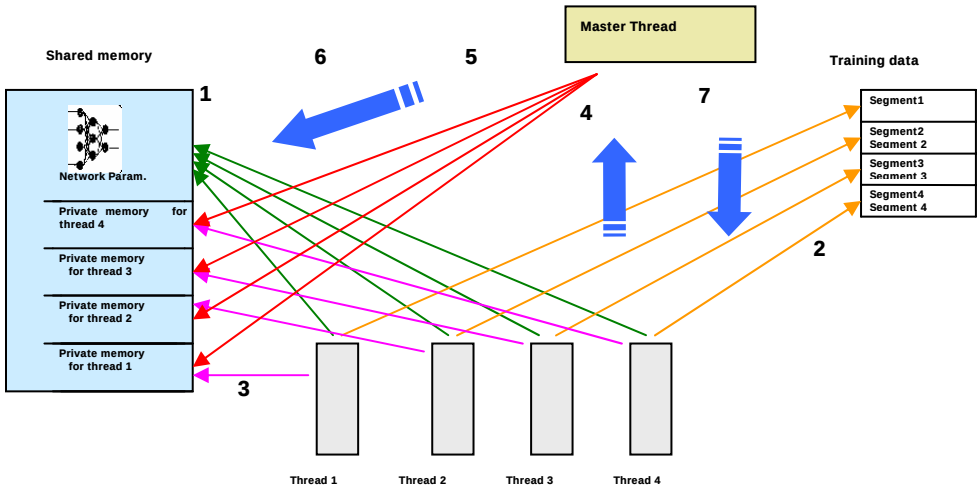

Figure 3. Implementation Details of Five Pthreads

Each thread is assigned to one of these sections

After one master and four slave threads are created, the master thread enters a waiting state. Then the following steps are followed:

Step 1: The slave threads read parameters of the K-means clustering algorithm from the shared memory. In this case, every thread has the same copy of the K-means clustering parameters.

Step 2: Each slave thread gets its portion of the training set. The training set is divided equally among all threads to establish a balanced workload.

Step 3: After calculating the errors, each slave thread updates its private memory space allocated for it in the shared memory. Every time a slave thread updates the memory space, it enters the waiting state.

Step 4: The last thread that updates the private memory space signals the main thread to wake up. After that, the slave thread enters the waiting state as well. Now, all slave threads are in the waiting state and doing nothing.

Step 5: The master thread wakes up upon receiving the wake up signal and reads the errors from the private memory space allocated for slave threads.

Step 6: The master thread updates the weight coefficients of the network.

Step 7: The master thread sends a broadcast signal to wake up all the slave threads. Upon sending this signal, the master thread enters the waiting state. The slave threads start the process from Step 1 again and the cycle goes on.

An example of the program codes for Pthread and OpenMP implementations are given in Figure 4 and Figure 5. The OpenMP has a much shorter code than Pthread because OpenMP hides the low-level details of iteration, space partitioning, data sharing, and thread scheduling

and synchronization from users. As a result, one simple command block can be used to create and synchronize different child threads under control of one master thread.

Child Thread

```
calculate(protos, beta, gamma, alpha,filearray[tid], tid);
pthread_mutex_lock(&count_mutex_cond2);
count=count+1;    /* signaling main thread*/
if (count == COUNT_LIMIT)
pthread_cond_signal(&count_threshold_cvp)
thread_cond_wait(&count_threshold_cond2, &count_mutex_cond2);
```

Parent Thread

```
pthread_cond_wait(&count_threshold_cv, &count_mutex_cond2);
calculateGradient(prevDprotos, prevDbeta, prevDgamma, prevDalpha, &ePrev);
pthread_cond_broadcast(&count_threshold_cond2);
```

Figure 4. Pthread Code

```
omp_set_num_threads(NUM_THREADS);
#pragma omp parallel private(nthreads, tid)
{
tid = omp_get_thread_num();  /* Obtain and print thread id */
calculate(protos, beta, gamma, alpha,filearray[tid], tid);
}
K-means clustering-update-function( );
```

Figure 5. OpenMP Code

4.5 Comparing Pthread and OpenMP Implementations

The compiler has built-in optimizations specific to Intel's Hyper-Threading architecture. It also integrates parallelization tightly with other advanced optimization techniques to achieve better cache locality and reduce the overhead of data sharing among threads (Tian et al., 2002). The speedup values for Pthread and OpenMP are presented in Figure 6. Pthread gives a higher

speedup value than that of OpenMP. This higher speedup ratio results from our neural network program implementation. When Pthread is used for parallelizing, the threads can be created only once and used many times with explicit synchronization among threads. However, when OpenMP is used for parallelization, the parallel region is created within the local function and the local function is called many times. Consequently, there is no way to keep the threads alive once the local function is returned and memory space allocated to local function is reclaimed. Therefore, all threads are destroyed after the return of the local function. Therefore, OpenMP loses performance efficiency by creating and destroying threads every time the local function is called. On the other hand, the same threads can be used repeatedly once threads are created in the Pthread implementation. Various coding techniques for the OpenMP program have been used to create threads outside the local function so that threads can persist during the execution of the program. However, these techniques only produce inconsistent results. This was one of the drawbacks in the OpenMP program. Although there is some communication overhead for the threads to signal each other in the Pthread implementation, this communication overhead is much less than the overhead produced by the thread creation and destruction of OpenMP. The advantage of the Pthread program produces a better speedup value.

When the number of threads continues to grow, the increasing cost of context switching and synchronization among the threads will decrease the efficiency of OpenMP and Pthread. As a result, the speedup will go down eventually after the number of threads becomes very big.

To speed up the training process, the K-means clustering algorithm is parallelized with Pthread and OpenMP. Higher speedup and lower execution time are reached when Pthread is used. Hyper-Threading technology is effective for the biological parallel algorithms and will be beneficial for future parallelization research. The parallel training program can make it possible

to process thousands of amino acids in a short amount of time in order to speed up tedious and intensive computational biomedical jobs.

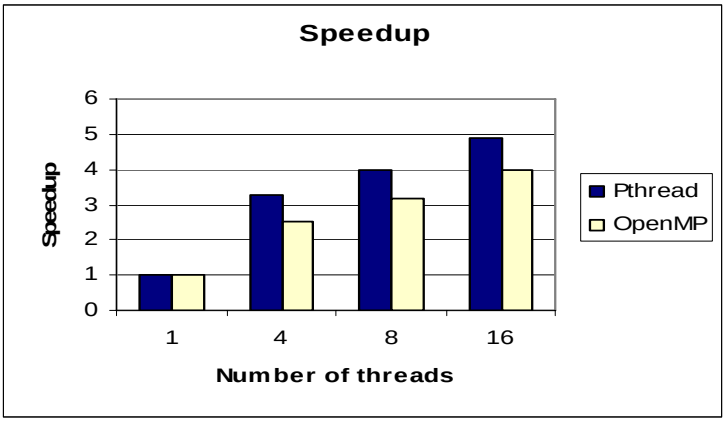

Figure 6. Speedup Values for Pthread and OpenMp

Chapter 5 Relationship between Sequence Variation and Corresponding Structural Similarity for Sequence Clusters and Sequence Motifs

In Chapter 3, an improved K-means clustering algorithm is proposed to discover the sequence clusters and sequence motifs automatically. In Chapter 4, parallel K-means clustering algorithm is used to speed up the clustering process. In this chapter, we want to discuss how sequence variation for sequence clusters may influence its structural similarity. How sequence variation for sequence clusters may influence its structural similarity is one of the most important tasks of current bioinformatics research.

Protein sequences are converted into sliding sequence segments. The sequence segments are classified into sequence clusters by the improved K-means clustering algorithm. No structural information is used during the clustering process. After the clustering is completed, the sequence variation for sequence clusters is analyzed by the number of important positions in the frequency profiles of sequence clusters. Furthermore, structural similarity for sequence clusters is assessed by secondary structural similarity and distance matrix root mean square deviation for sequence clusters (dmRMSD_SC). Analysis of the relationship between sequence variation and structural similarity for sequence clusters shows that sequence clusters with tight sequence variation have high structural similarity and sequence clusters with wide sequence variation have poor structural similarity. This finding has profound influence on building the protein grammar reflecting sequence-structure correspondence.

In this chapter, previous studies for sequence and structural variation of sequence clusters is reviewed first. Then recurrent clustering, data set and generation of sequence segments are introduced. Evaluation of sequence variation and structural similarity is discussed in details.

Finally, results analysis about the relationship between sequence variation and structural variation is presented.

5.1 Previous Studies for Sequence and Structural Variation of Sequence Clusters

Analysis of the relationship between the sequence variation and corresponding structural variation for sequence clusters is one of open questions for protein structure and sequence analysis (Rahman and Zomaya, 2005). Some researchers have evaluated the structural variation for sequence clusters. Kasuya and Thornton (1999) and Jonassen et al. (1999) have used cRMSD to analyze structural variation for sequence motifs. Bystroff and Baker (1998) have used the K-means clustering algorithm to find sequence clusters and to assess structural variation for these sequence clusters. Bystroff and Baker incorporated structural information during the clustering process (1998). As a result, final sequence clusters are contaminated by usage of structural information during the clustering process. Our implementation of the K-means clustering is significantly different from Bystroff's work (1998) because we only use recurrent clusters and do not include structural information in the clustering process. Meanwhile, Sander and Schneider (1991) have assessed sequence variation by the relative entropy for multiple sequence alignment. To the best of our knowledge, no researchers have conducted in-depth analysis of the relationship between sequence variation and corresponding structural variation for sequence clusters.

This work focuses on systematic and detailed analysis of the relationship between sequence variation and corresponding structural variation for sequence clusters (Zhong et.al, 2005a). During the process of generating sequence clusters, no structural information is used so that the true relationship between protein structure variation and sequence variation for sequence clusters can be accurately reflected. Understanding this relationship is very important to improve the

quality of local sequence alignment and low homology protein folding. Sequence clusters with tight sequence variation can be used to establish structural templates for low homology protein folding. Frequency profile of sequence clusters with tight sequence variation also can be used to find sequence segments with similar local structure in the local sequence alignment algorithm.

5.2 Experimental Setup

5.2.1 Recurrent Clustering

As introduced in Chapter 3, different number of initial clusters were tried and based on these results, 800 clusters are chosen empirically. Since the K-means clustering algorithm is sensitive to starting points, the numerical stability of the cluster algorithm is estimated by performing K-means clustering five times with different random starting points. Only recurrent clusters come into the final analysis of results.

5.2.2 Dataset

The dataset used in this work includes 2000 protein sequences obtained from the Protein Sequence Culling Server (PISCES) (Wang and Dunbrack, 2003). This training set is the same data set used in Chapter 9 for the improved K-means algorithm study. This data set is the training set for local protein structure prediction, which will be introduced in the later chapter. No sequences of this database share more than 25% identity. In other words, this database contains non-redundant protein sequences. The structures of these protein sequences are available from Protein Data Bank (Berman et al., 2002)

5.3 Clustering of Sequence Segments in the Sequence Space

Protein sequences are converted into sliding sequence segments with lengths ranging from 5 to 17 residues. The frequency profiles defined in the similarity-derived secondary structure of proteins (HSSP) (Sander and Schneider, 1991) are chosen as the numerical representation for sequence segments. At first, the sliding sequence segments with the length of five are classified into different clusters with the K-means clustering algorithm. No structural information is used during the whole clustering process. After the clustering algorithm is complete, sequence variation of sequence clusters having sequence segments with the length of five is studied. Furthermore, the structural variation of sequence clusters having sequence segments with the length of five is also assessed. The sequence segments with other lengths are similarly clustered and evaluated. The relationship between sequence variation and structural variation are studied for sequence clusters having sequence segments with specified lengths respectively.

5.4 Generation of Frequency Profile for Sequence Clusters

After the clustering is complete, the frequency profiles for sequence clusters having sequence segments with the specified length are produced. During the process of generating frequency profiles for sequence clusters, the frequency for the specified amino acid residue in a given window position for a cluster is calculated by division of the number of specified residues by the total number of residues in that position. For the window size of nine, there are nine positions in the frequency profiles and each position indicates the frequency of twenty amino acids in the sequence clusters.

5.5 Evaluation of Distribution of Amino Acid for Each Position of Frequency Profile

Sander and Schneider have used the relative entropy to describe the sequence variability (Sander and Schneider, 1991). In their calculation of the relative entropy, they did not consider the equilibrium frequency of amino acids. As a result, their assessment of the relative entropy is not accurate.

The relative entropy is used to describe the extent to which the distribution of 20 amino acids in the specified position of the frequency profile is uniform. The relative entropy measures the difference between the amino acid equilibrium distribution of amino acids in the database and the distribution of amino acids in the specified position of frequency profiles. Two distributions with more differences will result in a larger entropy value. In other words, larger entropy values reveal tight and increasingly imbalanced amino acid distribution in the specified position of the frequency profile and smaller entropy values represent increasingly uniform amino acid distribution in the specified position of the frequency profile.

Given the frequency of the amino acid of type R at the specified position of the frequency profile and the equilibrium frequency P_R of the amino acid of type R, the relative entropy in the specified position of frequency profiles is defined as (Bebiano, 2005):

$$RE = \sum_{R}^{20} f_R \ln \frac{f_R}{P_R} \quad (3)$$

where the equilibrium frequency P_R of the amino acid of type R is calculated by division of the total number of amino acids of type R by the total number of amino acids in the database.

5.6 Measure of Sequence Variation for a Given Sequence Cluster

Sequence variation for a given sequence cluster has been evaluated by two different measures. The effectiveness of these two measures has been compared in this work.

5.6.1 Measure of Sequence Variation by Average of Relative Entropy Values for All Position of Sequence Profiles

The first measure to evaluate sequence variation is to calculate the average of relative entropy values for all positions of the sequence profile. Results show that the average of relative entropy values for all positions does not distinguish the sequence clusters with high structural variation and with low structural variation.

5.6.2 Measure of Sequence Variation by the Number of Important Positions for Sequence Profiles

Since the average of relative entropy values for all positions of frequency profiles does not work, we need to consider the distribution of amino acids in each position of frequency profile individually.

If the relative entropy in the specified position of the frequency profiles is greater than 0.2, this position is defined as the important position for frequency profiles. Our statistics indicate that an average of five amino acids occupy 60% of the frequency space if the relative entropy in that position of the frequency profiles is greater than 0.2. Statistically, each of twenty amino acids may occur with the frequency of 5%. Therefore, five amino acids may occupy 25% of the frequency space. As a result, the distribution of amino acids is highly disproportionate in the important positions.

The number of important positions is used to indicate the extent of sequence variation for sequence clusters. Increased number of important positions in the frequency profiles reflects more positions in the frequency profiles have highly disproportionate distribution of 20 amino acids. As a result, sequence variation for sequence clusters is more compact. In contrast, relatively small number of important position indicates the sequence variation for sequence clusters is wide. Our results indicate that defining sequence variation for sequence clusters by the

number of important position is more effective in distinguishing the sequence clusters with high structural variation and low structural variation. Thus, the results presented in this work are based on the measure of sequence variation by the number of important positions of sequence profiles.

5.7 Measure of Secondary Structure Similarity for a Given Sequence Cluster

After the clustering process is completed, the structural variation of sequence-based clusters is evaluated by secondary structure similarity and dmRMSD_SC.

In this paper, H represents helices; E represents sheets and C represents coils. The higher secondary structure similarity reveals lower structural variation for sequence clusters. The formula 4 is used to calculate secondary structural similarity for a given cluster (Han and Baker, 1996):

$$\frac{\sum_{i=1}^{L} \max(P_{i,H}, P_{i,E}, P_{i,C})}{L} (\%) \quad (4)$$

L is the length of sequence segments. P(i,H) is the frequency of occurrence of helices among the sequence segments for the cluster in position i. P(i,E) and P(i, C) are similarly defined.

5.8 Measure of Tertiary Structural Variation by dmRSMS_SC for A Given Sequence Cluster

dmRMSD_SC represents the average value of dmRMSD between the distance matrix of each sequence segment and its average distance matrix for a sequence cluster. Smaller dmRMSD_SC values indicate that the distance matrices for sequence segments are closer to their ADM and sequence segments for one sequence cluster have tighter structural variation.

5.8.1 Average Distance Matrix (ADM) among Sequence Segments for a Given Sequence Cluster

ADM records the average for the distance matrices of all the sequence segments in one sequence cluster, using the formula 5:

$$\alpha_{i\to j}^{ADM} = \frac{\sum_{k=1}^{N} \alpha_{i\to j}^{k}}{N} \quad (5)$$

where N is the number of sequence segments of a given cluster.

5.8.2 dmRMSD_SC for a Given Sequence Cluster

The formula 6 is used to calculate dmRMSD_SC:

$$dmRMSD\_SC = \frac{\sum_{k=1}^{N} \sqrt{1/M \sum_{i=1}^{L} \sum_{j=i+1}^{L} \left(\alpha_{i\to j}^{k} - \alpha_{i\to j}^{ADM}\right)^2}}{N} \quad (6)$$

$$M = \frac{(L \times L - L)}{2} \quad (7)$$

where $\alpha^{s1}{}_{i \to j}$ is the distance between α-carbon atom i and α-carbon atom j in the sequence segment s1 of the length L and M is the number of distances in the distance matrix.

5.9 Results Analysis

The sequence variation and structural variation of sequence clusters having sequence segments with the specified length are analyzed separately. The length of sequence segments ranges from 5 to 15 in our study. Sequence clusters having sequence segments with different lengths show the similar relationship between sequence variation and structure variation for sequence clusters. The frequency profile for each cluster is considered one word in our grammar system. Analysis of the dependence of provides the important foundation to establish words of various lengths for sequence-based vocabulary for protein structure. In this work, we consider

sequence clusters containing sequence segments with the length of nine. All the results shown in the following are related to the sequence clusters having sequence segments with the length of nine.

Figure 7 shows the relationship between variability and the relative entropy for each position of sequence profiles (Zhong et.al, 2004b). Variability indicates the number of amino acids occurring with the frequency greater than its equilibrium frequency in the specified position of the sequence profile for a given sequence cluster. Figure 7 indicates that the variability of different positions in the frequency profiles for sequence clusters decreases as values for relative entropy increases. This pattern is reasonable since the increasing relative entropy represent stronger conservation in that position of frequency profiles. As a result, Figure 8 supports that the relative entropy is the good measure to define amino acid distribution patterns in the specified positions of frequency profiles.

Figure 8 shows the percentage of clusters with the specified number of important positions in the specified ranges of secondary structure similarity. The percentage of clusters with two important positions and secondary structural similarity between 60% and 70% is calculated by division of the number of clusters with two important positions and secondary structural similarity between 60% and 70% by the total number of clusters having secondary structural similarity between 60% and 70%. The X-axis gives the number of important positions. As the number of important position increases, the percentage of clusters with secondary structure similarity between 50% and 60% shrinks rapidly. In contrast, the percentage of clusters with secondary structure similarity between 80% and 100% shrinks slowly. The number of important positions for clusters with secondary structure similarity between 80% and 100% is greater than four. Meanwhile, the majority of sequence clusters with secondary structural similarity between

50% and 60% have the important positions less than four.

Figure 9 compares the important positions between the percentage of clusters with dmRMSD_SC > 2.0 Å and the percentage of clusters with dmRMSD_SC < 1.5 Å. The clusters with dmRMSD_SC > 2.0 Å have secondary structure similarity less than 50% and are clusters with widest tertiary structural variation. The clusters with dmRMSD_SC < 1.5 Å have secondary structure similarity greater than 72% and are clusters with tightest tertiary structural variation. The majority of clusters with dmRMSD_SC > 2.0 Å have two important positions. In contrast, the majority of clusters with dmRMSD_SC < 1.5 Å have more than five important positions.

Analysis of Figure 8 and Figure 9 reveals that on average, the number of important positions for clusters with low structural variation is greater than the number of important positions for clusters with high structural variation. Low structural variation for sequence clusters indicates that structural variation is compact. A large number of important positions indicate that sequence variation for sequence clusters is tight. In other words, Figure 8 and figure 9 show an important pattern that sequence clusters with tight sequence variation typically have tight structural variation and sequence clusters with wide sequence variation typically have wide structural variation.

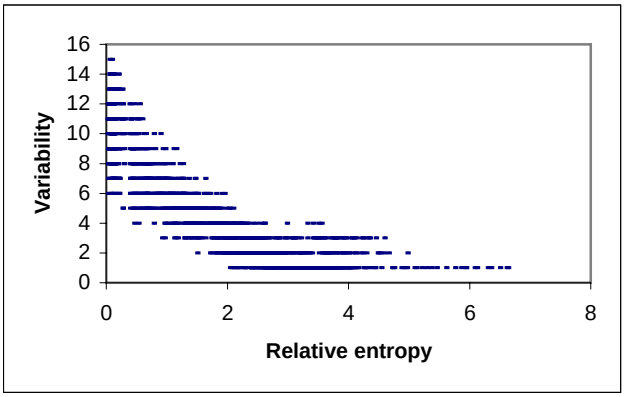

Figure 7 Relationship between Variability and the Relative Entropy for Each Position of Sequence Profiles for Sequence Cluster

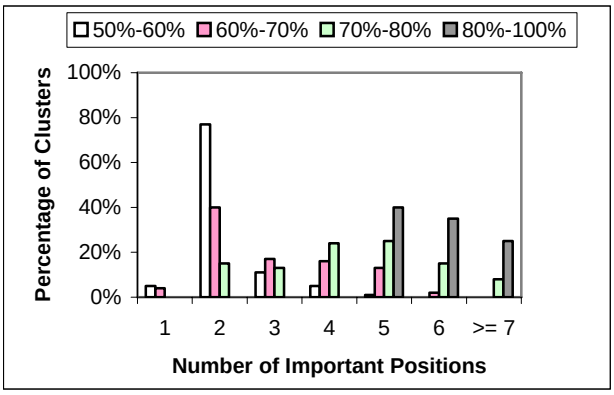

Figure 8 Percentages of Sequence Clusters with the Specified Number of Important Positions in the Specified Ranges of Secondary Structure Similarity

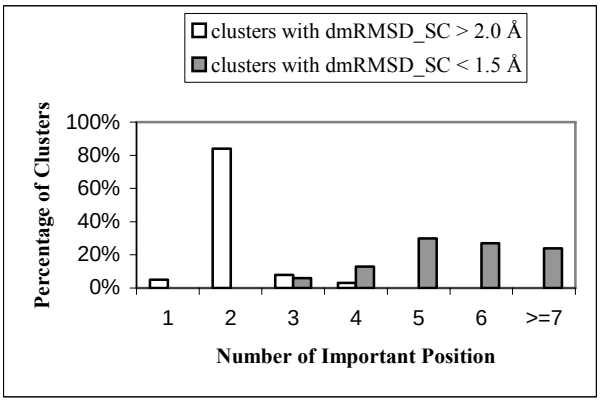

Figure 9 Comparison of the Important Positions between the Percentage of Clusters With dmRMSD_SC > 2.0 Å and the Percentage of Clusters With dmRMSD_SC < 1.5 Å

Our study indicates the important relationship between sequence variation and structure variation for sequence clusters. This work also shows that there is a sequence-based vocabulary for protein structure. This sequence-based vocabulary can be used to develop the protein grammar showing sequence-structure correspondence. Due to time constraints and incomplete information in the database, words in the vocabulary will have multiple meanings.

Chapter 6 Local Protein Structure Prediction by the Clustering System

In Chapter 3 and Chapter 5, we have discussed the improved K-means algorithm for sequence motif discovery and how sequence variation for sequence clusters may influence its structural similarity. Based on above knowledge, the clustering system is used for local protein structure prediction.

Protein structure prediction is one of the open problems of computational biology today. Knowing the structure of a protein sequence enables us to probe the function of the protein, to perform drug design, and to construct novel proteins. Determination of protein structure can also provide important information for various researches such as mapping the functions of proteins in metabolic pathways for whole genomes. In this chapter, carefully constructed clustering system are used to predict local protein structure.

In this chapter, how to cluster sequence segments into clusters is explained first. Then the method to calculate the representative structure for each cluster is explained. Distance score and reliability score to decide the cluster membership is discussed. The performance evaluation and experimental results are explained in the last part of this chapter.

6.1 Data Set and Sequence Segment Generation

6.1.1 Training Set and Independent Test Set

The training data set used in our work includes 2000 protein sequences obtained from the Protein Sequence-Culling Server (PISCES) (Wang and Dunbrack, 2003). 200 protein sequences from the recent release of PISCES are included into the independent test set. For the comparative purpose, the training set and testing set is same as that for clustering support vector machines

introduced in Chapter 9. The structures of these protein sequences are available from Protein Data Bank (PDB) (Berman et al., 2002). Any two sequences in the training set and the test set share less than 25% similarity.

6.1.2 Clustering of Sequence Segments Belonging to the Training Set

The method for generating sequence segments is same as that introduced in Chapter 3. The sliding sequence segments are generated from protein sequences in the training set. These sequence segments are grouped into different clusters by the K-means clustering algorithm. During the process of generating sequence clusters, no structural information is used. The frequency profile defined in the similarity-derived secondary structure of proteins (HSSP) (Sander and Schneider, 1991) is chosen as the numerical representation for sequence segments. This work focuses on a sequence segment of nine. Our preliminary results show that the sequence segments with the length of nine are long enough to have some structural features and are short enough to have a statistically significant number of samples. It is clear that other segment lengths are important and the analysis presented here can be applied to them as well. Due to huge amount of computation, we plan to analyze the sequence segments from the length ranging from 5 to 15 in the next step.

6.2 Representative Structure for Each Cluster

The representative structure of each cluster is represented by average distance matrix, representative torsion angle and representative secondary structure form.

6.2.1 Representative Secondary Structure

In this paper, H represents helices; E represents sheets and C represents coils. The following formula is used to calculate the average level of secondary structure similarity among sequence

segments for a given cluster (Han and Baker, 1996):

$$\frac{\sum_{i=1}^{L} \max(P_{i,H}, P_{i,E}, P_{i,C})}{L} \quad (8)$$

L is the length of sequence segments. P(i,H) is the frequency of occurrences of helices among the sequence segments for the cluster in position i. P(i,E) and P(i,C) are similarly defined. The secondary structure with the maximum frequency is used for representing secondary structure for the cluster.

6.2.2 Average Distance Matrix (ADM)

Average Distance Matrix (ADM) records the average for the distance matrices of all the sequence segments in one cluster, using the formula 9:

$$\alpha_{i \to j}^{ADM} = \frac{\sum_{k=1}^{N} \alpha_{i \to j}^{k}}{N} \quad (9)$$

where $\alpha_{i \to j}^{k}$ is the distance between α-carbon atom i and α-carbon atom j in the sequence segment k of the length L. N is the total number of sequence segments in the cluster.

6.2.3 Representative Torsion Angle

Torsion angles range between –180˚ and 180˚. In this work, we propose the new formula to calculate the modular distanace of torsion angles. The modular distance makes sure that the maximum difference between two torsion angles is not greater than 180˚. The following formula is used to calculate the modular distance (mod_dis) between two torsion angles:

$$\text{mod\_}dis(a,b) = \min(abs1, abs2, abs3) \quad (10)$$

$$abs\,1 = |a - b| \quad (11)$$

$$abs\,2 = |(a + 360\,^\circ) - b| \quad (12)$$

$$abs\,3 = |a - (b + 360\,^\circ)| \quad (13)$$

where a and b are two torsion angles and the modular distance (mod_dis) is the minimum value of abs1, abs2 and abs3. a and b are ϕ or ψ defined in (Karp, 2002).

ϕ_i is the representative ϕ in the ith position of sequence segments for sequence clusters. All the values in the position i of sequence segments in a sequence cluster are put into a set. The representative ϕ_i is defined as the ϕ value that is taken from this set and has the minimum sum of modular distances to the other members of this set. In a sense, ϕ_i is the center of this set. ψ_i is similarly defined.

6.3 Local Structure Prediction by the Clustering System

In this section, we explain how to predict local structures of protein sequences based on distance scores and reliability scores of the clustering system. As described previously, the sliding windows with nine successive residues are generated from protein sequences. Each window represents one sequence segment. Structure of each sequence segment is predicted by the rule-based system.

6.3.1 Distance Score of a Given Sequence Segment for Each Cluster

The frequency profile for a given sequence segment is compared with the centroid of the each cluster in order to calculate the distance score. The distance score is used to filter out some less significant clusters. A smaller distance score shows that the frequency profile of the given sequence segment is closer to the centroid for a given cluster. The centroid of the given cluster is the average of all frequency profiles of sequence segments for this cluster.

The following formula calculates the distance score of a given sequence segment for a given cluster (Han and Baker, 1996).

$$Distance\_score = \sum_{i=1}^{L} \sum_{j=1}^{N} |F_k(i, j) - F_c(i, j)| \quad (14)$$

$$z - score = \frac{Score - Average}{Std} \quad (15)$$

where L is the window size and N is 20. $F_k(i, j)$ is the value of frequency profile at row i and column j for the sequence segment k. $F_c(i, j)$ is the value of the matrix at row i and column j for the centroid of this cluster. Average is the average of scores from all sequence segments for this cluster. Std is the standard deviation of scores from all sequence segments for this cluster.

6.3.2 Reliability Score of Each Cluster for a Given Sequence Segment

The reliability score of a given sequence segment for a cluster is determined by the sum of the frequency of the matched amino acid in the corresponding position of the average frequency profile of a cluster. Higher reliability scores indicate that prediction results by this cluster is more dependable since the amino acids of the given sequence segment match more frequently occurring amino acids in the corresponding position of a cluster for structure conservation.

$$\text{Reliability score} = \sum_{i=1}^{L} F_c(i, j) \quad (16)$$

where $F_c(i, j)$ is the value of the matrix at row i and column j for the average frequency profile of the cluster. The value of j is determined by the type of amino acid in the specified position of sequence segment.

6.3.3 Structure Prediction by Distance Score and Reliability Score for a Given Sequence Segment

The distance score value of each cluster for a given sequence segment is calculated in order to filter out some less significant cluster. If the difference of the cluster's distance score and the smallest distance score is within 100, this cluster is selected. Other clusters are discarded since

they are less significant. The cluster with the highest reliability score among the selected clusters finally functions to predict the structure of this sequence segment.

The distance score efficiently narrows down the list of possible clusters based on similarity of the frequency profile for the given sequence segment and the centroid of this cluster. The reliability score assesses how well the amino acids of a given sequence segment match key amino acids in the important positions in order to conserve a particular local structure. Our prediction results shows that the combination of the distance score and the reliability score can improve the prediction accuracy of the clustering system noticeably since the distance score and the reliability score carry very independent information.

6.4 Prediction Accuracy Calculation

Accuracy for structure prediction of sequence segments in terms of secondary structure accuracy, dmRMSD and taRMSD is calculated to evaluate the performance of the clustering system. Distances between α-carbon atoms and backbone torsion angles are two important structural constraints for representing protein 3D structure. As a result, this comprehensive evaluation scheme including dmRMSD and taRMSD is used to assess the prediction results.

6.4.1 Secondary Structure Accuracy

Q3 is one of the most commonly used performance measures in the protein secondary structure prediction. Q3 refers to the three-state overall percentage of correctly predicted residues. The following formula is used to calculate secondary structure accuracy (Hu et. al. 2004):

$$Q3 = \frac{\sum_{i\in\{H,E,C\}} \#\ of\ residues\ correctly\ predicted_i}{\sum_{i\in\{H,E,C\}} \#\ of\ residues\ in\ class\ i} \quad (17)$$

6.4.2 Distance Matrix Root Mean Square Deviation (dmRMSD)

The following formula is used to calculate dmRMSD (Kolodny and Linial, 2004; Zagrovic and Pande, 2004):

$$dmRMSD = \sqrt{\frac{\sum_{i=1}^{L}\sum_{j=i+1}^{L}\left(\alpha_{i\to j}^{s1} - \alpha_{i\to j}^{ADM}\right)^2}{M}} \quad (18)$$

$$M = \frac{(L \times L - L)}{2} \quad (19)$$

where $\alpha_{i\to j}^{ADM}$ is the distance between α-carbon atom i and α-carbon atom j in the average distance matrix of a rule. M is the number of distances in the distance matrix in this formula.

6.4.3 Torsion Angle RMSD (taRMSD)

$$taRMSD = \sqrt{\frac{\sum_{k=1}^{L}\left\{\left(\phi_{ki} - \phi_{kj}\right)^2 + \left(\psi_{ki} - \psi_{kj}\right)^2\right\}}{2L}} \quad (20)$$

where ϕ_{kj} is ϕ in the position k of the representative angle for a rule and ψ_{kj} is ψ in the position k of the representative angle for a rule. ϕ and ψ are defined in (Karp, 2002).

6.4.4 Classification of Clusters into Different Groups

During the prediction process, structures of sequence segments are first predicted by clusters with the high training accuracy. If the structures of sequence segments cannot be predicted by clusters with high training accuracy, clusters with the lower training accuracy will be used for structure prediction.

Training secondary structure accuracy for a given cluster is the average training accuracy of sequence segments in the training set predicated by this cluster. Training dmRMSD of a given cluster is the average training dmRMSD of sequence segments in the training set predicated by this cluster. Training taRMSD of a given cluster is similarly defined. Test secondary structure

accuracy, test dmRMSD and test taRMSD is similarly defined for each cluster in the independent test set.

Table 3 shows the standard to classify clusters into different groups based the training accuracy of the clustering algorithm. In the good cluster group, all clusters have training secondary structure accuracy greater than 80%, training dmRMSD less than 1 Å and training taRMSD less than 25 degree. The bad cluster group and the average cluster group are similarly defined. As a result, the good cluster group includes all the clusters with highest training accuracy. The bad cluster group includes clusters with poor training accuracy. In this work, thirty clusters are randomly chosen from each cluster group to test the performance of the prediction system.

Table 3 Standard to Classify Clusters into Different Groups

	Secondary Structure Accuracy	dmRMSD	taRMSD
Bad Cluster Group	between 60% and 70%	greater than 1.5 Å	greater than 30 degree
Average Cluster Group	between 70% and 80%	between 1 Å and 1.5 Å	between 25 and 30 degree
Good Cluster Group	greater than 80%	less than 1 Å	less than 25 degree

6.4.5 Accuracy criteria for Each Cluster

Only combined information of secondary structure, torsion angle and distance matrix can represent protein structure precisely. In order to rigorously evaluate the prediction quality for these algorithms, we used two sets of accuracy criteria named accuracy criteria one and accuracy criteria two. Accuracy criteria one and accuracy criteria two consider secondary structure accuracy, dmRMSD and taRMSD simultaneously. Table 4 provides the threshold for evaluating accuracy criteria one and accuracy criteria two for each cluster. Accuracy criteria two for one cluster is the percentage of sequence segments with secondary structure accuracy greater than

80%, dmRMSD less than 1 Å and taRMSD less than 25 degree in the test set for this cluster. Accuracy criteria two reflect the percentage of sequence segments with the most reliable structure prediction for one cluster. Accuracy criteria one are similarly defined. An accuracy criteria one reflects the percentage of sequence segments with acceptable level of structure prediction for one cluster.

Table 4 the Threshold for Evaluating Accuracy Criteria One and Accuracy Criteria Two for Each Cluster

	Secondary Structure Accuracy	dmRMSD	taRMSD
Accuracy Criteria One	> 70%	< 1.5 Å	< 30 degree
Accuracy Criteria Two	> 80%	< 1 Å	< 25 degree

6.5 Expreimental Results

Figure 10 compares the secondary structure accuracy between different cluster groups for the clustering system. Secondary structure accuracy for the bad cluster group is 65.13%. Secondary structure accuracy for the average cluster group reaches 74.02%. Secondary structure accuracy for the good cluster group is 82.10%.

Figure 11 compares dmRMSD between different cluster groups for the clustering system. The dmRMSD error for the bad cluster group reaches 1.92 Å. The dmRMSD error for the average cluster group reduces by 26% compared to the bad cluster group. The dmRMSD error for the good cluster group reduces by 46% compared to the bad cluster group.

Figure 12 compares the taRMSD between different cluster groups for the clustering system. The taRMSD error for the bad cluster group reaches 52.34 degree. The taRMSD error for the

average cluster group reduces by 21% compared to the bad cluster group. The taRMSD error for the good cluster group reduces by 38% compared to the bad cluster group.

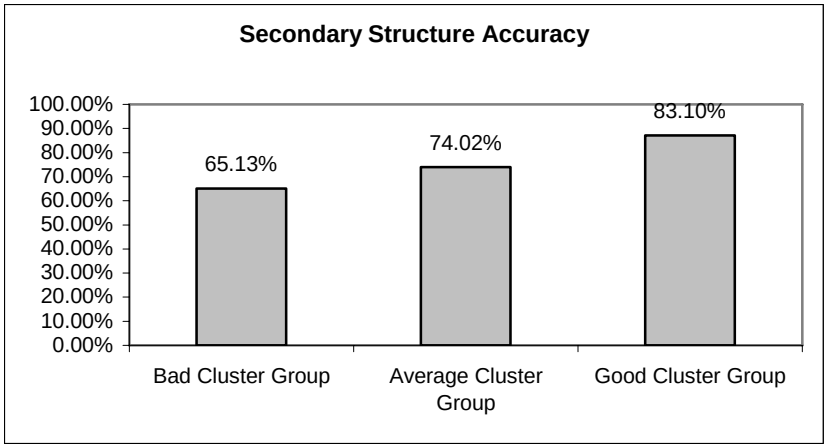

Figure 10 Secondary Structure Accuracy for the Clustering System

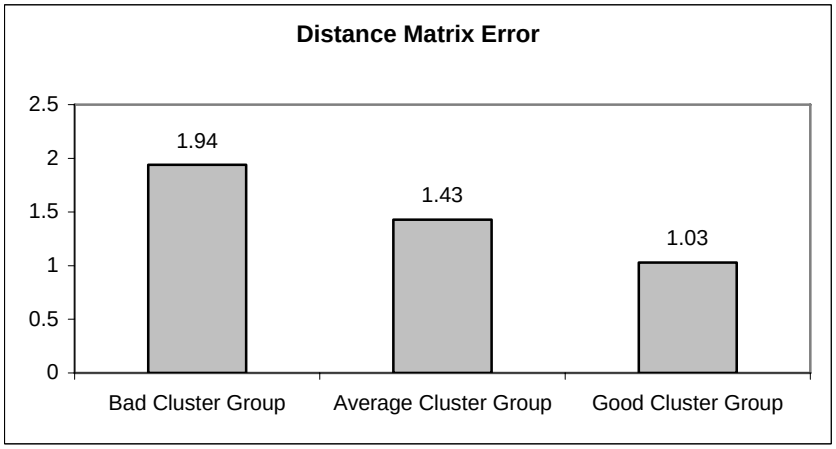

Figure 11 dmRMSD for the Clustering System

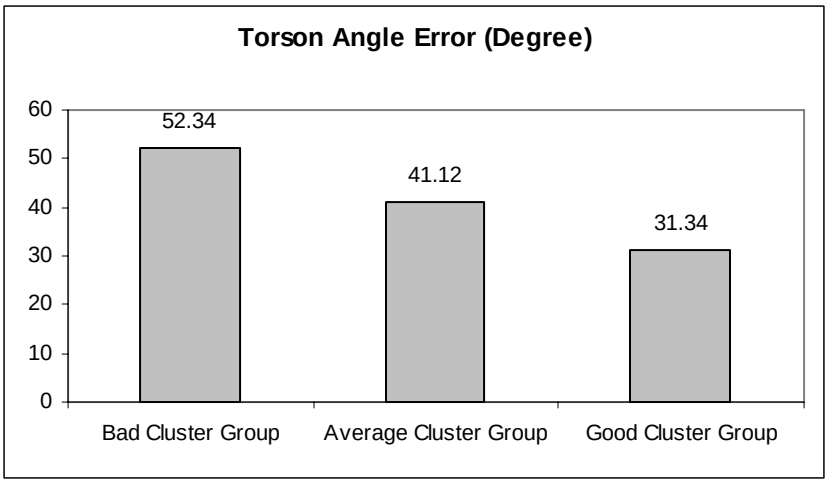

Figure 12 taRMSD for the Clustering System

As described previously, accuracy criteria one and accuracy criteria two for local protein structure prediction have considered three evaluation metrics including secondary structure accuracy, dmRMSD and taRMSD simultaneously. Since three metrics reflect the prediction accuracy in different perspectives, consideration of three metrics together will give the most rigorous evaluation for the quality of structure prediction. Accuracy criteria one reflects the percentage of sequence segments whose structural prediction is acceptable. Accuracy criteria two indicates the percentage of sequence segments whose structural prediction is the most reliable. Figure 13 compares accuracy criteria one between different cluster groups for the clustering system. Figure 14 compares accuracy criteria two between different cluster groups for the clustering system. Figure 13 shows that accuracy of the good group cluster has improved by 17% compared to the bad cluster group in terms of accuracy criteria one. Figure 13 shows that accuracy of the good group cluster has improved by 27% compared to the bad cluster group in terms of accuracy criteria two. All these figures indicate that clusters with high quality provide the reliable prediction results and clusters with average quality produces high quality results. Special cautions need be taken against prediction results from the bad cluster group.

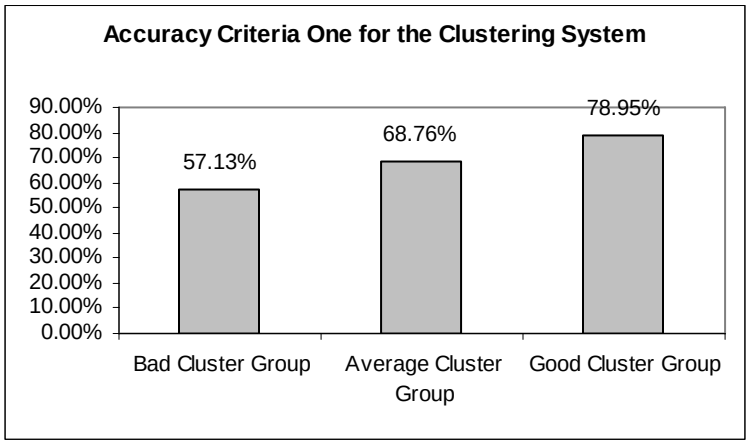

Figure 13 Accuracy Criteira One for the Clustering System

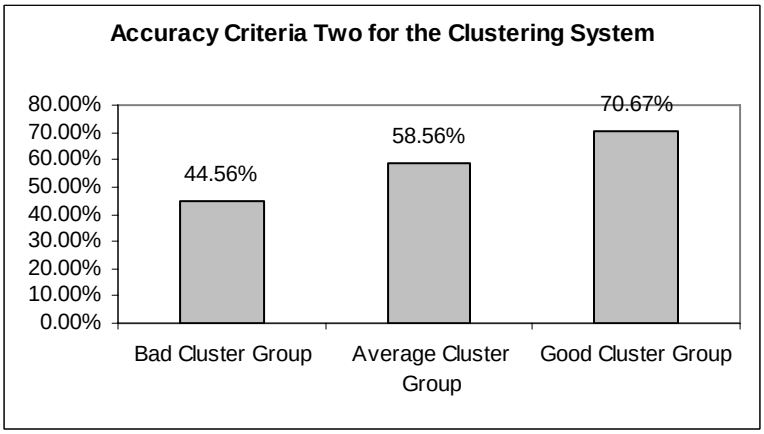

Figure 14 Accuracy Criteira Two for the Clustering System

After the clustering system for local protein prediction is explained in detailed in the second part of dissertation, the clustering support vector machine is proposed in order to improve the performance of the clustering system. In the third part of the dissertation, the foundational information about Support Vector Machine is explained first. Then various methods to solve the problem of large datasets training for SVM are discussed. Finally the motivation and major steps of Clustering Support Vector Machines is given.

Chapter 7 Support Vector Machine

Support Vector Machines are a new generation of learning machines, which have been successfully applied to a wide variety of application domains (Cristianini and Shawe Taylor, 2000) including bioinformatics (Schoelkopf, Tsuda and Vert, 2000). Construction of optimal hyperplane that can separate samples belonging to the first class fr om samples belonging to the second class with the maximal margin is the essential task of SVM. In this chapter, the concept of optimal hyperplane and optimization problems to construct optimal hyperplane in the linearly separable case and in the linearly nonseparable case will be discussed first. Then the expected risk bounds are evaluated to assess the effectiveness of support vector machines. Also the quadratic optimization and linear optimization method to build SVMs are discussed. SVM Kernels play key roles in calculating the inner products between support vectors and the vectors implicitly in the high dimensional feature space, several important SVM kernels are introduced in this section. In real world, we need solve multiclassification problem besides two-class classification. Multiple classifications for SVM are also explained.

7.1 Optimal Hyperplane for Separable Case

In this section, the detailed process of solving optimization problems is explained in order to construct hyperplane, which can separate data points linearly. After solving the optimization, one specific algorithm is introduced to implement the ideas of solving the optimization problem.

7.1.1 Optimization Problem to Build Optimal Hyperplane

We need find a pair consisting of ψ_0 and a constant b_0 to satisfy the following constraints (Vapnik, 1998):

$$y_i((x_i * \psi_0) + b) \geq 1, \quad i = 1,..., l \quad (21)$$

and the vector ψ_0 has the smallest norm $|\psi|^2 = (\psi * \psi)$

ϕ_0 is the vector deciding the optimal hyperplane. ϕ_0 is defined by equation 22

$$\phi_0 = \frac{\psi_0}{|\psi_0|} \quad (22)$$

The margin ρ_0 between separating hyperplane and separated vectors is equal to

$$\rho(\phi_0) = \sup_{\phi_0} \frac{1}{2}\left(\min_{i \in I} (x_i * \phi_0) - \max_{j \in II} (x_j * \phi_0) \right) = \frac{1}{|\psi_0|} \quad (23)$$

The vector ψ_0 with the smallest norm satisfying constraints 21 with b =0 defines the optimal hyperplane passing through the origin (Vapnik, 1998).

In order to find optimal hyperplane, we need solve the quadratic optimization problem by minimizing the quadratic form $|\psi|^2 = (\psi * \psi)$ under the linear constraints. We can use Lagrange multipliers defined by the equation 24 to solve the quadratic optimization problems (Vapnik, 1998).

$$L(\psi, b, \alpha) = \frac{1}{2}(\psi * \psi) - \sum_{i=1}^{l} \alpha_i \left(y_i [(x_i * \psi) + b] - 1 \right) \quad (24)$$

where $\alpha_i \geq 0$ are the Lagrange muttipliers.

In order to find the saddle point we need minimize the function over ψ and b and to maximize it over the nonnegative Lagrange multipliers $\alpha_i \geq 0$. After minimizing the function over ψ and b, we produce the equation 25 and the equation 26 (Vapnik, 1998).

$$\psi = \sum_{i=1}^{l} y_i \alpha_i x_i \quad (25)$$

$$\sum_{i=1}^{l} y_i \alpha_i = 0 \quad (26)$$

Taking into account the above two equations, we obtain the equation 27 which is transformed from the equation 24 (Vapnik, 1998).

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j (x_i * x_j) \quad (27)$$

In order to construct optimal hyperplane, we need find α_i^0 that maximize the margin in the nonnegative quadrant considering the constraint 26. Using coefficient $\alpha_i^0, i = 1,..., l$, we can obtain the equation 28 based on the equation 25.

$$\psi_0 = \sum_{i=1}^{l} y_i \alpha_i^0 x_i \quad (28)$$

The value of b_0 is chosen to maximize margin.

Ψ_0 and b_0 need satisfy the Kuhn-Tucker conditions

$$\alpha_i^0 \left(y_i ((x_i * \psi_0) + b_0) - 1 \right) = 0, \qquad i = 1,\ldots l. \quad (29)$$

The vectors with nonzero α_i^0 are those data points closest to the optimal hyperplane. These vectors are called support vectors. The support vectors are key to construct the hyperplane since ψ_0 defining the optimal hyperplane depends on nonzero weights on support vectors. As a result, the optimal hyperplane has the form 30 (Vapnik, 1998).

$$f(x, \alpha_0) = \sum_{i=1}^{l} y_i \alpha_i^0 (x_s * x) + b_0 \quad (30)$$

where x_s is the support vectors.

Since the separating hyperplane defined by the equation 29 and the optimization problem defined by the equation 27 do not depend on the dimensionality of the vector x, this will help us to construct hyperplane in the high dimensional feature space.

7.1.2 Some Properties of Hyperplane and One Algorithm to build Optimal Hyperplane

In this section, some properties of hyperplane and one specific algorithm to construct hyperplane are discussed in details.

The maximum of the functional W(α) is equal to the equation 31(Vapnik, 1998).

$$W(\alpha) = \frac{1}{2}(\psi_0 * \psi_0) = \frac{1}{2}\sum_i \alpha_i^0 \quad (31)$$

The margin of the optimal separating hyperplane is determined by the norm of the vector ψ_0 (Vapnik, 1998).

$$\rho(\psi_0) = \frac{1}{|\psi_0|} \quad (32)$$

From the equation 31 and 32, we derive that

$$W(\alpha) < W(\alpha_0) = \frac{1}{2}\left(\frac{1}{\rho(\psi_0)}\right)^2 \quad (33)$$

In order to maximize the functional W(α), the number of the support vectors and the coefficient α need to be determined (Vapnik, 1998). In the first step, small number of samples is selected with their corresponding coefficient as nonzero. After the value W(α) is maximized, nonzero coefficient α is kept and new parameters are added. New parameters are associated with vectors, which cannot be separated properly by the hyperplane constructed in the first iteration. These process will continue until all the training data are separated or $W(\alpha) > W_{max}$. In this algorithm, the function of W(α) is maximized depending on part of data sets which are candidates of support vectors (Vapnik, 1998).

7.2 Optimal Hyperplane for Nonseparable Sets

After explaining how to construct the hyperplane in the linearly separable case, the method to construct the hyperplane in nonseparable case is discussed.

7.2.1 Δ-margin Separating Hyperplanes

For the data, which cannot be linearly separated, the concept of Δ-margin separating hyperplane is introduced. A hyperplane is a Δ-margin separating hyperplane defined by the equation 34 (Vapnik, 1998):

$$(\psi^* * x) - b = 0, \quad |\psi^*| = 1 \quad (34)$$

if the following constraints are satisfied

$$y = \begin{cases} 1 & if \quad (\psi^* * x) - b \geq \Delta \\ -1 & if \quad (\psi^* * x) - b \leq \Delta \end{cases} \quad (35)$$

In order to build Δ-margin separating hyperplane with nonlinear separable case, the function 36 is introduced (Vapnik, 1998):

$$F(\xi) = \sum_{i=1}^{l} \xi_i \quad \xi_i \geq 0 \quad (36)$$

We need minimize the functional F(ξ) subject to constraints 37 and 38

$$y_i((\psi * x_i) - b \geq 1 - \xi_i, \quad i = 1,2,..., l \quad (37)$$

$$(\psi * \psi) \geq \Delta^{-2} \quad (38)$$

The hyperplane with parameters that minimize functional $F(\xi)$ subject to above constraints.

To solve the optimization problem, we introduce the following saddle point of Lagrangian (Vapnik, 1998):

$$L(\psi, b, \alpha, \beta, \gamma) = \sum_{i=`}^{l} \xi_i - \frac{1}{2}\gamma(A^2 - (\psi * \psi)) - \sum_{i=1}^{l} \alpha_i(y_i((\psi * x_i) + b) - 1 + \xi_i) - \sum_{i=1}^{l} \beta_i \xi_i \quad (39)$$

After minimizing Lagrangian with respect to ψ, b, ξ, the equation 40, equation 41 and equation 42 are produced (Vapnik, 1998).

$$\psi = \frac{1}{\gamma}\sum_{i=1}^{l} \alpha_i y_i x_i \quad (40)$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0, \qquad (41)$$

$$\alpha_i + \beta_i = 1 \qquad (42)$$

Substituting the equation 40 into the Lagrange, we obtain the function 43 (Vapnik, 1998).

$$W(\alpha, \gamma) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2\gamma} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j (x_i * x_j) - \frac{\gamma A^2}{2} \quad (43)$$

We need maximize the function 43 under the constraints

$$\sum_{i=1}^{l} \alpha_i y_i = 0, \quad (44)$$

$$0 \le \alpha_i \le 1, \qquad (45)$$

$$\gamma \ge 0 \qquad (46)$$

We can first solve the quadratic optimization problem several times for fixed values of γ and maximize Lagrange with respect of these γ values. When the maximum γ is reached, the equation 47 need to be satisfied (Vapnik, 1998).

$$\lambda = \frac{\sqrt{\sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j (x_i * x_j)}}{A} \quad (47)$$

The generalized optimal hyperplane with parameters $\alpha_0 = \left(\alpha_1^0, ..., \alpha_l^0\right)$ is defined by the equation 48 (Vapnik, 1998).

$$f(x) = \frac{A}{\sqrt{\sum_{i,j=1}^{l} \alpha_i^0 \alpha_j^0 y_i y_j (x_i * x_j)}} \sum_{i=1}^{l} \alpha_i^0 y_i (x * x_i) + b \quad (48)$$

7.2.2 Soft Margin Generalization

The following slightly modified concept of the generalized optimal hyperplane is introduced. In order to obtain optimal hyperplane, the function 49 is minimized with the respect of ψ under the constraints $F(\xi) = \sum_{i=1}^{l} \xi_i \quad \xi_i \ge 0$ (Vapnik, 1998).

$$\phi(\psi, \xi) = \frac{1}{2}(\psi * \psi) + C\left(\sum_{i=1}^{l} \xi_i\right) \quad (49)$$

In order to obtain the optimal hyperplane, the quadratic form is optimized using the formula 50 (Vapnik, 1998):

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j (x_i * x_j) \quad (50)$$

under constraints:

$$0 \le \alpha_i \le C, \quad i = 1,..., l, \quad (51)$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \quad (52)$$

Vectors with nonzero coefficients form the support vectors. The support vector and corresponding coefficient determines the optimal hyperplane (Vapnik, 1998).

$$\sum_{i=1}^{l} \alpha_i^0 y_i (x_i * x) + b_0 = 0 \quad (53)$$

7.3 Expected Risk Bounds for Optimal Hyperplane

After discussing how to construct the hyperplane in linearly separable case and linearly nonseparable case, the expected risks bounds for optimal hyperplanes are evaluated. The expected risk bound for the optimal hyperplane is smaller than the expected risks obtained from minimizing empirical risk.

The expected risk bound is defined based on the concept of essential support vectors. Essential support vectors are those support vectors who appears in all the possible expansion of the optimal hyperplane. In other words, it is the joint set of all possible sets of support vectors. The training set is denoted by $(x_1, y_1),..., (x_l, y_l)$. The number of essential support vectors is denoted by the equation 54 (Vapnik, 1998).

$$\kappa_l = \kappa\left((x_1, y_1),...(x_l, y_l)\right) \quad (54)$$

The maximum norm of the vector x from the set of essential support vectors is

$$D_l = D\left((x_1, y_1),..., (x_l, y_l)\right) = \max_i |x_i| \quad (55)$$

The expected risk bound for optimal hyperplane based on the training samples with the size of l is defined by the equation 56 (Vapnik, 1998)

$$ER(\alpha_l) \leq \frac{E\kappa_{l+1}}{l+1} \quad (56)$$

7.4 Mercer's Theorem to Deal with High Dimensionality

In this section, the fundamental concept of SVM is discussed. In addition, the Mercer's theorem, which is the key for solving high-dimensional mapping problems, is explained.

7.4.1 Fundamental Concept of SVM

The central ideas of support vector machines are to map the input space into another higher dimensional feature space using the kernels function and to build an optimal hyperplane in that feature space (Vapnik, 1998). For example, the kernel function including a polynomial of degree two can be used to map input vectors with n coordinates into the feature space with $\frac{n(n+3)}{2}$ coordinates.

7.4.2 Mercer's Theorem for High Dimensionality

One important question is how to build the hyperplane that has strong generalization capability in the high dimensional feature space. A second question is how to avoid "the curse of dimensionality" in this high dimensional feature space. Mercer's Theorem helps us avoid mapping the input space into another higher dimentional space explicitly (Vapnik, 1998).

If the vector $x \in R^n$ is mapped into a Hilbert space with coordinates $z_1(x),\ldots z_n(x),\ldots$, the inner product in a Hilbert space has this equivalent form (Vapnik, 1998):

$$(z_1 * z_2) = \sum_{r=1}^{\infty} \alpha_r z_r(x_1) z(x_2) \Leftrightarrow K(x_1, x_2), \alpha_r \geq 0, \quad (57)$$

where $K(x_1,x_2)$ is the inner product in some feature space.

Mercer's theorem indicates that any kernel function satisfying Mercer's condition can calculate the inner product of two vectors in some high dimensional Hilbert space (Vapnik, 1998). Based on Mercer's theorem, the high-dimensional feature space need not be considered directly during the process of finding the optimal hyperplane. Instead, the inner products between support vectors and the vectors in the feature space can be calculated.

7.5 Construction of SVM

In this section, the quadratic optimization and linear optimization method to build SVMs are discussed.

7.5.1 Constructing SVM with Quadratic Optimization

The equation 58 and 59 are linear and nonlinear functions building the optimal hyperplane. Nonlinear decision function 58 built from high dimensional feature space (Vapnik, 1998).

$$f(x, \alpha) = sign\left[\sum_{SV} y_i \alpha_i^0 K(x, x_i) + b\right] \quad (58)$$

The equation 59 is the equivalent linear decision functions in the high dimensional feature space $z_1(x),\ldots z_n(x),\ldots$,

$$f(x, \alpha) = sign\left(\sum_{SV} y_i \alpha_i^0 \sum_{r=1}^{\infty} z_r(x_i) z_r(x) + b\right) \quad (59)$$

In order to construct optimal hyperplane in the high-dimensional feature space, the inner product defined by kernel $K(x,x_i)$ can be used to replace the inner product defined in (x,x_i) (Vapnik, 1998).

To construct the optimal hyperplane in the separable case, we need maximize the following function 60 (Vapnik, 1998):

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad (60)$$

subject to the constraints

$$\sum_{i=1}^{l} \alpha_i y_i = 0, \quad (61)$$

$$\alpha_i \geq 0, \quad i = 1,2,..., l \quad (62)$$

To construct the optimal hyperplane in the nonseparable case using optimal soft margin solution, we need maximize the function 60 subject to the constraints (Vapnik, 1998):

$$\sum_{i=1}^{l} \alpha_i y_i = 0, \quad (63)$$

$$0 \leq \alpha_i \leq C \quad (64)$$

Complexity of Support Vector Machine is determined by the number of support vectors instead of the dimensionality of feature space. SVM has two layers. In the first layer, input vectors are implicitly transformed based on support vectors and each inner product between the input vector and support vectors are calculated based on the Kernel function (Vapnik, 1998). In the second layer, the linear decision function is built in the high dimensional feature space (Vapnik, 1998).

7.5.2 Constructing SVM using Linear Optimization Method

Since the optimal hyperplane expanded on the support vectors and the generalization ability of the constructed hyperplane depends on the number of support vectors, the following linear optimization method rather than quadratic optimization can be used (Vapnik, 1998).

Given the training data, we need find the parameters α_i and b of the hyperplane, which can satisfy the inequalities $y_j\left(\sum_{i=1}^{l} y_i\alpha_i(x_i * x) + b\right) \geq 1$ and has the smallest number of nonzero coefficients α_i.

To solve the optimization problem using linear optimization approaches, we need minimize the functional 65 (Vapnik, 1998):

$$R = \sum_{i=1}^{l} \alpha_i, \quad \alpha_i \geq 0 \qquad (65)$$

subject to constraints

$$y_j\left(\sum_{i=1}^{l} y_i\alpha_i(x_i * x) + b\right) \geq 1 \quad (66)$$

To build optimal hyperplane in the nonseparable case using linear optimization approaches, we need minimize the function 67 (Vapnik, 1998):

$$R = \sum_{i=1}^{l} \alpha_i + C\sum_{i=1}^{l} \xi_i, \quad \alpha_i \geq 0, \xi_i \geq 0 \quad (67)$$

over the nonnegative variable α_i, ξ_i and parameter b subject to the constraints 68

$$y_i\left(\sum_{j=1}^{l} \alpha_i(x_i * x_j) + b\right) \geq 1 - \xi_i, \quad i = 1,..., l \quad (68)$$

7.6 SVM Kernels

SVM Kernels play key roles in calculating the inner products between support vectors and the vectors implicitly in the high dimensional feature space, several important SVM kernels are introduced in this section.

7.6.1 Selection of SV Machine Using Bounds

Let $z(x) = (z_1(x),..., z_N(x),...)$ be data points in the feature space and let $w = (w_1(x),..., w_N(x),...)$ be a vector weights determining a hyperplane in this space.

Let us consider a set of hyperplanes containing the functions satisfying the conditions 69 (Vapnik, 1998).

$$\lfloor D^2 |w|^2 \rfloor \leq k \quad (69)$$

where D is the radius of the smallest sphere containing the vector ψ(x) and |w| is the norm of the weights. K provides the estimation of the VC dimension of the set of functions defined on the training set.

If the Support Vector can separate the training data without errors and has the minimal norm |w|, the Support Vector Machine has the smallest estimates of the VC dimension (Vapnik, 1998).

In order to minimize the expected error and control the generalization ability of the SV machines, the function 70 need be minimized (Vapnik, 1998):

$$R(D_l, w_l) = D_l^2 |w_l|^2 = \frac{D_l^2}{\rho_l^2} \quad (70)$$

where D_l and w_l can be calculated from the training set.

The best SV machine with the smallest expected risks has smallest VC dimension (Vapnik, 1998). In order to obtain the SV machine having small VC dimension, the function 70 is used to evaluate the upper bound of the VC dimension. Experimental experience shows that the SV

machine with small VC dimension does not necessarily map input vectors onto small dimensions in the feature space (Vapnik, 1998).

7.6.2 Polynomial Functions

The following kernel can be used to construct polynomial of degree d decision rule (Vapnik, 1998):

$$K(x, x_i) = [(x * x_i) + 1]^d \quad (71)$$

Using this kernel, we can construct a decision function of the form (Vapnik, 1998):

$$f(x, \alpha) = sign\left(\sum_{SV} y_i \alpha_i [(x_i * x) + 1]^d + b \right) \quad (72)$$

Polynomials of degree d will map input vectors in the n-dimensional input space into the feature with $O(n^d)$ coordinates. The VC dimension of the subset of polynomials for the real world problems can be low, resulting in low errors.

7.6.3 Radial Basis Functions

Classical radial basis function (RBF) machines uses the set of decision rules (Vapnik, 1998):

$$f(x) = sign\left(\sum_{i=1}^{N} \alpha_i K_\gamma (|x - x_i|) - b \right) \quad (73)$$

where $K_r(|x - x_i|)$ is based on the distance $|x - x_i|$ between two vectors. This is one type of these functions (Vapnik, 1998):

$$K_\gamma (|x - x_i|) = \exp\left\{ - \gamma |x - x_i|^2 \right\} \quad (74)$$

These functions are a positive definite monotonic function. In order to construct the above decision rules, we need determine the number N of the centers x_i, the vector x_i for centers, the values of the parameters α_i and γ_i.

In the classical RBF method, the heuristics method decides the parameter γ, the number of centers and centers x_i. Since the radial basis function satisfies the condition of Mercer's theorem, $K_r(|x - x_i|)$ can be selected to produce the inner product in some feature space. In order to build a SV radial basis function machine, the number of support vectors, the support vector x_i, the coefficients of expansion and the width parameter γ of the kernel function are chosen automatically (Vapnik, 1998).

7.6.4 Two-layer Neural Networks

This is the kernel defining two-layer neural networks (Vapnik, 1998):

$$K(x, x_i) = S[(x * x_i)] \quad (75)$$

$$S[(x * x_i)] = \frac{1}{1 + \exp\{v(x * x_i) - c\}} \quad (76)$$

where S(u) is a sigmoid function.

Based on the above kernel, the two-layer neural SV machine is constructed (Vapnik, 1998).

$$f(x, \alpha) = sign\left\{\sum_{i=1}^{N} \alpha_i S(v(x * x_i) - c) + b\right\} \quad (77)$$

During the optimization process, the number of hidden units, vectors of the weights in neurons of the first hidden layer and the vector of weights for the second layer will be determined automatically.

7.7 Multiclass Classification

In the real world, we need to solve multiclassification problems besides the two-class classification. SVM can also solve the multiclassification problem. We can construct n-class classifiers based on two-class classification. In the first step, n two-class classification rules, which separate training samples of one class from other training samples, are constructed. In the

second step, n-class classifiers are built from selecting the class corresponding to the maximal value of n two-class classifiers as indicated in the equation 78 (Vapnik, 1998).

$$finalclass = \arg\max \{f_1(x_l),..., f_n(x_l)\} \quad (78)$$

where n two-class classifiers are indicated by $f_k(x_i)$, k = 1,…,n

Chapter 8 Implementation of SVM for a Very Large Dataset

SVMs are based on the idea of mapping data points to a high dimensional feature space where a separating hyperplane can be found. SVMs are searching the optimal separating hyper-plane by solving a convex quadratic programming (QP). The typical running time for the convex quadratic programming is Ω (m^2) for the training set with m samples. The convex quadratic programming is NP-complete in the worst case (Vavasis, 1991). Therefore, SVMs are not favorable for a large dataset (Chang and Lin, 2001). Our dataset contains a half million samples. Our experiments show that training of SVM has not completed for the half million samples after one month on the "poweredge6600 server" with four processors from Dell®. According to Hwanjo Yu, Jiong Yang, and Jiawei Han (2003), it would take years to train SVMs on a data set containing one million records.

Many algorithms and implementation techniques have been developed to enhance SVMs in order to increase their training performance with large data sets. The most well known techniques include chunking (Vapnik, 1998), Osuna's decomposition method (Osuna, Freund, and Girosi, 1997), Sequential Minimal Optimization (SMO) (Platt, 1999) and boosting algorithms (Pavlov, Mao and Dom, 2000). The success of these methods depends on dividing the original quadratic programming (QP) problem into a series of smaller computational problems in order to reduce the size of each QP problem. Although these algorithms accelerate the training process, these algorithms do not scale well with the size of the training data.

The second class of algorithms tries to speed up the training process by reducing the number of training data. Since some data points such as the support vectors are more important to determine the optimal solution, these algorithms provide SVMs with high quality data point. During the training process, Random Selection (Balcazar, Dai and Watanabe, 2001), active

learning clustering (Scholhn and Cohn, 2000) and clustering analysis (Yu, Yang, and Han, 2000) are representatives of these algorithms. These algorithms are highly scalable for the large data set while the performance of training depends greatly on the selection of training samples.

In this chapter, the algorithms dividing the original quadratic programming (QP) problem into a series of smaller computational problems is discussed first. Then the second class of algorithms trying to speed up the training process by reducing the number of training data is explained.

8.1 First Class of Algorithms for Large Dataset

The success of these methods depends on dividing the original quadratic programming (QP) problem into a series of smaller computational problems in order to reduce the size of each QP problem. There are several algorithms have been proposed to speed up the training speed, including chunking (Vapnik, 1998), Osuna's decomposition method (Osuna, Freund, and Girosi, 1997), Sequential Minimal Optimization (SMO) (Platt, 1999) and boosting algorithms (Pavlov, Mao and Dom, 2000). Although these algorithms have been proven to speed up the training process, they do not scale well with the size of the training data.

8.1.1 Decomposition Algorithm

During the training process of SVM, the linearly constrained Quadratic Programming (QP) problem with a number of variables equal to the number of data points will be solved. This problem becomes very challenging when the size of data set grows very large. Previous problems assume that the number of support vectors is small compared to the number of data points and the total number of support vectors does not exceed a few thousands since the ratio between the number of support vectors and the total number of data points is the upper bound on

the generalization error. However, the ratio between the number of support vectors and the total number of data points is high and the data set is very large in many difficult problems. Even if a problem has small generalization errors, the number of support vectors can still be large.

The decomposition problem proposed by Osuna (Osuna, Freund, and Girosi, 1997) will not make certain assumption on the expected number of support vectors and enable us to train SVM on a large data set by solving a sequence of smaller QP problems. Optimality conditions and the strategy for improving optimization goals are two key issues in this algorithm. The optimality conditions are essential to make sure the objective function can improve at each iteration under the decomposition strategy. A large QP problem can be broken down into a series of smaller QP subproblems. As long as at least one example that violates the KKT condition is included into the dataset for subproblems, each step will move towards the final goal of the objective function while maintaining all of the relevant constraints.

In Osuna's decomposition algorithm (Osuna, Freund, and Girosi, 1997), the optimization problem is divided into an inactive and an active part. In this decomposition strategy, the variables α_i of the optimization problem are divided into the set B of free variables and the set N of fixed variables. Free variables can be updated in the current iteration and fixed variables are temporarily fixed at a particular value. In each step, q variables for the working set is selected and the remaining variables are fixed at current value. The optimization subproblem on the set B is performed. If the optimality conditions are not satisfied, the algorithm decomposes the optimization problem and solves the smaller QP-problems. The decomposition makes sure that the process are moving towards the final goal of maximizing the object function if the working set B meets the minimum requirements (Osuna, Freund, and Girosi, 1997). This iteration will repeat until optimality conditions are satisfied. In this decomposition algorithm, the memory

requirement is linear in the number of training examples and is linear in the number of support vectors. However, this algorithm may need a long training time.

8.1.2 Sequential Minimal Optimization (SMO)

Chunking is proposed by Vapnik to solve the optimization problem (Vapnik, 1998). If the rows and columns of the matrix correspond to zero, Lagrange Multiplier can be removed and the quadratic form remains the same. At every step, chunking solves the optimization problems including every non-zero Lagrange multiplier from the last step and m worst examples that violate the KKT conditions. Each QP subproblem is based on the results of the previous subproblem. At the end of iteration, the entire set of non-zero Lagrange multipliers have been identified. Chunking seriously reduces the size of the computation matrix to approximately the number of non-zero Lagrange multipliers squared. However, chunking cannot deal with large-scale training problems because the reduced computation matrix still cannot fit into the memory.

In order to solve the memory problem, Sequential Minimal Optimization (SMO) is proposed by Platt (1999). During training a support vector machine, a very large quadratic programming problem needs to be solved. SMO divides the large QP problem into a series of smallest possible QP problems (Platt, 1999). These small QP problems are solved without using the time-consuming numerical QP optimization as an inner loop. SMO scales between linear and quadratic in the training set size for several test problems (Platt, 1999).

SMO selects the smallest possible optimization problem at every step. In the SVM's QP problem, the smallest possible optimization problem has two Lagrange multipliers. At each step, SMO chooses two Lagrange multipliers to jointly optimize, finds the optimal values for these multiples and adds updated values to SVM. Since two Lagrange multipliers can be optimized analytically, numerical QP optimization can be avoided entirely. Even though more optimization

subproblems need to be solved, each subproblem is so fast that the overall QP problem can be solved quickly. An analytic method for solving the optimization problem related to the two Lagrange multiplies and a heuristic for choosing which multipliers to optimize are two major research issues for SMO.

For SMO, there are two separate heuristics methods. The first heuristic method is used to select the first Lagrange multiplier, which provides the outer loop of the SMO algorithm. The outer loop will check all samples and determine whether each sample violates KKT conditions. If an example violates the KKT conditions, it is eligible for optimization. Once the first Lagrange multiplier is chosen, SMO will choose the second Lagrange multiplier to maximize the size of the step taken during joint optimization.

The SMO can be considered a special case of the Osuna algorithm, where the size of the optimization is two (Platt, 1999). Both Lagrange multipliers are replaced at every step with new multipliers selected by heuristic methods. Since SMO treats linear SVMs in a special way, it can speed up the training process for linear separators. Unlike other algorithms, SMO uses the smallest possible QP problems, which can be solved analytically. Solving QP problems analytically can improve the computation time quickly.

8.1.3 Boosting Algorithm to Scale up SVM

Pavlov, Mao and Dom have proposed to solve the scaling problems of SVM by the boosting algorithm (Pavlov, Mao and Dom, 2000). Boosting can potentially convert a weak classifier into a strong classifier, which can obtain strong generalization ability given enough training data. In this work, the scaling problem of SVM is solved with comparable testing accuracy. Boosting will focus on errors made by the previous iteration during training a sequence of classifiers. During the boosting procedure, each training sample is assigned a probability label and is maintained

over the whole training phase. If a particular example is misclassified by the previously built classifier, this example will be given a higher probability.

Subsamples of the boosting set are produced based on the probability distribution. The boosting set for training the classifier on the t_{th} iteration can be produced by selecting samples from the original data set based on data distribution. The size of the boosting set is roughly equal to 2-4% of the original set which will increase the training speed substantially. Since the boosting algorithm can improve the margin of hyperplanes, combination of boosting and SMO can increase the training speed while finding a global solution, which is comparable in terms of accuracy to that obtained by the standard SVM training algorithm.

For the boosted classifier, we need first choose the appropriate size of training individuals for SVM (Pavlov, Mao and Dom, 2000). Typically, the fewer number of boosting steps are required with larger subset size. However, larger subset size may reduce the training speed of the individual SVM. The number of boosting steps need be determined. In certain cases, smaller boosting steps may lead to better generalization performance.

8.2 Second Class of Algorithm for Large Dataset Training

The first class of algorithms discussed in the section 8.1 divides the original QP problem into small pieces and reduces the size of each QP problem. No theoretical guarantee has been given on the efficiency of algorithms based on these techniques. In this section, the second class of algorithms is discussed. The second class of algorithms tries to speed up the training process by reducing the number of training data. Since some data points such as the support vectors are more important to determine the optimal solution, these algorithms provide SVMs with high quality data points during the training process. Random selection (Balcazar, Dai and Watanabe,

2001), active learning clustering (Scholhn and Cohn, 2000) and clustering analysis (Yu, Yang, and Han, 2000) are representatives of these algorithms.

8.2.1 Random Selection

The scalability properties show that we can possibly bring the SVM methodology to a very large dataset. However, the performance of SVM deteriorates in case of having many outliers compared with the dimensionality of the data. As a result, the random selection algorithm for training SVM is proposed by Balcazar, Dai and Watanabe (2001). In this approach, small number of samples is randomly selected by repeatedly filtering the selection through a probability distribution that evolves according to the results of the previous phases. The upper bound on the expected running time is quasilinear on the number of data points (Balcazar, Dai and Watanabe, 2001).

The randomized subset selection scheme can be applied to dual form, which is key to take advantages of a major feature of support vector machine (Balcazar, Dai and Watanabe, 2001). If the number of samples is much larger than the dimension n, the randomized sampling techniques are effective. In this random sampling technique, a small number of samples from the large dataset are selected under the set of constraints to these samples. Samples are selected randomly according to their weights. Initially all samples are given the same weight. In the obtained local solutions, some samples are misclassified. Weights of misclassified examples are double. After this process has been repeated for several times, the weight of important samples, which are support vectors, grow exponentially fast (Balcazar, Dai and Watanabe, 2001). After all support vectors are selected at certain iteration, the local solution becomes true global solution.

8.2.2 Active Learning with SVM

With the active learning heuristic method, a SVM trained on a well-chosen subset of data samples performs better than the SVM trained on all available data. This active learning algorithm can provide good generalization ability and requires fewer data than a passive learner trained on the entire data (Scholhn and Cohn, 2000). In the selective sampling, the learner will choose to label some subset of large amount of unlabeled examples. In a probabilistic framework, an active learner can estimate the expected future error and can select training examples that are expected to minimize this expected future error.

Platt (1999) describes a greedy optimal strategy to assign probabilities to points in the space. In the first step, all examples are projected onto an axis perpendicular to the dividing hyperplane and logistic regression is performed on them to extract class probabilities. By integrating the probability of errors weighted by some assumed distributions of test examples over the volume of the space, the expected error of the classifier can be estimated. However, this algorithm is impractical since evaluating each candidate point requires solving two QP problems.

In order to reduce the computation time of the greedy optimal strategy developed by Platt (1999), a simple heuristic is developed to estimate the expected change in error from adding an example without recomputing SVM (Scholhn and Cohn, 2000). It is assumed that samples that lie along the dividing hyperplane will divide the space up most quickly. A data point's location will have strong effect on how the data point will be labeled. Labeling a sample lining on or close to the hyperplane will influence the solution strongly. Selection of training samples by their distance to the dividing hyperplane is computationally inexpensive. If the dividing hyperplane can be computed explicitly, evaluating each candidate requires only a single dot product computation. An active learner starts with a small training set and iteratively increases its size

with comparative computational performance to chunking. An active learner can minimize number of labels for non-support vectors since they have no effect on formation of deciding hyperplane (Scholhn and Cohn, 2000).

8.2.3 Classifying Large Datasets using SVM with Hierarchical Clusters

Despite the prominent feature of SVM, SVM is not favorable for large-scale data mining because the training complexity of SVMs is highly dependent on the size of a dataset. Many data mining applications will have millions of samples, which make SVM training impractical since simple scanning may take a long time. Clustering-based SVM (CB-SVM) is produced to speed up the training process (Yu, Yang, and Han, 2000). A hierarchical micro-clustering algorithm will scan the entire data set in order to provide an SVM with high quality samples that carry the statistical summaries of the data. These statistical summaries will increase efficiency of the SVM learning process. With increasing size of samples, performance of SVM trained on entire data set is worse that that of SVM trained on intelligently selected data sample. CB_SVM is scalable in terms of the training efficiency while maximizing the performance of SVM.

In the CB-SVM algorithm, two micro–clustering trees for positive samples and negative samples are built respectively (Yu, Yang, and Han, 2000). In each tree, the node in a higher level is the summarized representation of child nodes. CB-SVM will start to train the SVM from the root node. After obtaining the rough boundary from the root node, CB-SVM will selectively decluster the data based on the hierarchical representation. CB-SVM is effective especially when the random sampling deteriorates the performance because of infrequently occurring important boundary data.

After single scans of the database, clustering trees provide a statistically summarized representation of a group of data, which are likely to belong to one cluster. The clustering feature

tree may capture the major distribution patterns of the data and may provide enough information for SVM training. It also handles outliers, which are not part of the underlying distribution effectively. The clustering feature tree is a compact representation of the data set since each entry in a leaf node is subclusters of data points with similar characteristics.

In each iteration, CB-SVM selects the low margin data which is close to the boundary in the feature space since the low margin data have better chances to become the VCs of the boundary for the next round. In order to realize this idea, the entries near the boundary are declustered in order to get finer samples nearer to the boundary and courser samples farther from the boundary. Using this strategy, the data, which has high probability to become the support vector, can be introduced to the training while keeping total training size small.

The detailed procedure of CB-SVM is discussed here. Two CF-trees are constructed from positive and negative samples. SVM is trained from the centroids of the root entries for two CF trees. The entries near the boundary will be declustered into the next level. Children entries declustered from the parent entries are accumulated into the training set with the non-declustered parent entries. New SVM is constructed from the centroids of entries in the training set. In this algorthm, the CF-tree provides suitable structure to perform the selective declustering efficiently. The clustered data provides better summaries for SVM than random samples of the entire data set. Random sampling may be susceptible to a biased input and produces undesirable results.

Many heuristics can speed up SVM training by dividing the original QP problem into small pieces in order to reduce the size of QP problems. Chunking, decomposition and sequential minimal optimization are most well-known examples. CB-SVM can reduce the size of training set by converting data into the statistical summaries of large data groups. The course summary is used for unimportant data, which is far away from the decision boundary. Fine summary is used

for important data. The important data is close to the decision boundary and have high potential to become support vectors. Since different techniques to solve the large dataset training have their disadvantages, a new computation model called CSVMs is proposed in the next chapter.

Chapter 9 Clustering Support Vector Machines for Protein Local Structure Prediction

Understanding sequence-to-structure relationship is a central task in bioinformatics research. Adequate knowledge about this relationship can potentially improve accuracy for local protein structure prediction. In Chapter 6, one of approaches for protein local structure prediction has been introduced. In these approaches, the conventional clustering algorithms are used to capture the sequence-to-structure relationship. The cluster membership function defined by conventional clustering algorithms may not reveal the complex nonlinear relationship adequately. Compared with the conventional clustering algorithms, Support Vector Machine (SVM) can capture the nonlinear sequence-to-structure relationship by mapping the input space into another higher dimensional feature space. However, SVM is not favorable for huge datasets including millions of samples. Therefore, we propose a novel computational model called CSVMs (Clustering Support Vector Machines). Taking advantage of both theory of granular computing and advanced statistical learning methodology, CSVMs are built specifically for each information granule partitioned intelligently by the clustering algorithm. This feature makes learning tasks for each CSVM more specific and simple. CSVMs modeled for each granule can be easily parallelized so that CSVMs can be used to handle complex classification problems for huge datasets. Average accuracy for CSVMs is over 80%, which indicates that the generalization power for CSVMs is strong enough to recognize the complicated pattern of sequence-to-structure relationships. Compared with the clustering system introduced in Chapter 6, our experimental results show that accuracy for local structure prediction has been improved noticeably when CSVMs are applied.

This chapter is organized as follows. In the section 9.1, previous research is reviewed. In the section 9.2, the advantage of granular computing and SVM is introduced. A new computational model called Clustering Support Vector Machines is also discussed in detail. In the section 9.3, the training set, the testing set and accuracy definition are explained. In the section 9.4, the experimental results and analysis are given. Finally, the conclusion and the future work are presented.

9.1 Review of Previous Work

Understanding protein sequence-to-structure relationship is one of the most important tasks of current bioinformatics research. The knowledge of correspondence between the protein sequence and its structure can play very important role in protein structure prediction (Rahman and Zomaya, 2005). Many biochemical tests suggest that a sequence determines conformation completely, because all the information that is necessary to specify protein interaction sites with other molecules is embedded into its amino acid sequence (Karp, 2002). These studies form the experimental basis for exploring the relationship between the protein sequence and its structure.

Han and Baker have used the K-means clustering algorithm to explore protein sequence-to-structure relationship. Protein sequences are represented with frequency profiles. With the K-means clustering algorithm, high quality sequence clusters have been produced (Han and Baker, 1996). They have used these high quality sequence clusters to predict the backbone torsion angles for local protein structure (Bystroff and Baker, 1998). In 2000, they set up Hidden Markov Model (HMM) based on high quality sequence clusters and used HMM to predict the backbone torsion angles for local protein structure (Bystroff, Thorsson and Baker, 2000). In their work, the K-means clustering algorithm is essential to understand how protein sequences correspond to local 3D protein structures. However, the conventional clustering algorithms such

as the K-means and K-nearest neighbor algorithm assume that the distance between data points can be calculated with exact precision. When this distance function is not well characterized, the clustering algorithm may not reveal the sequence-to-structure relationship effectively. As a result, some of clusters provide poor correspondence between protein sequences and their structures.

Support Vector Machine (SVM) are new generation of machine learning techniques and have shown strong generalization capability for many data mining tasks. SVM can handle the nonlinear classification by implicitly mapping input samples from the input feature space into another high dimensional feature space with the nonlinear kernel function. Therefore, SVM may be more effective to reveal the nonlinear sequence-to-structure relationship than K-means clustering does. The superior performance for non-linear classification inspires us to explore the relationship between the protein sequence and its structure with SVM.

Since SVM is not favorable for a large dataset (Chang and Lin, 2001) as introduced in Chapter 8, modeling of one SVM over the whole feature space containing almost half million data samples is impractical. Furthermore, each subspace of the whole feature space corresponds to different local 3D structures in our application. As a result, construction of one SVM for the whole feature space cannot take advantage of the strong generalization power of SVM efficiently. The disadvantage of building one SVM over the whole feature space motivates us to consider the theory of granular computing. Using the divide-and-conquer principle, granular computing is able to divide a complex data-mining problem into a series of smaller and computational simpler problems (Yao, 2005).

To combine the theory of granular computing and principles of the statistical learning algorithms, we propose a new computational model called CSVMs (Clustering Support Vector

Machines) in our work. In this new computational model, one SVM is built for each information granule defined by sequence clusters created by the clustering algorithm. CSVMs are modeled to learn the nonlinear relationship between protein sequences and their structures in each cluster. SVM is not favorable for large amount of data samples. However, CSVMs can be easily parallelized to speed up the modeling process. After gaining the knowledge about the sequence to structure relationship, CSVMs are used to predict distance matrices, torsion angles and secondary structures for backbone α-carbon atoms of protein sequence segments. Compared with the clustering system introduced in Chapter 6, CSVMs can estimate how close frequency profiles of protein sequences correspond with local 3D structures by using the nonlinear kernel. Introduction of CSVMs can potentially improve the accuracy of local protein structure prediction.

9.2 Method

In this section, the principle of granular computing and SVM is introduced. Explanation of the motivation to combine the granular computing and SVM will provide deeper understanding about advantages of the new computational model. The procedures to train CSVMs modeled for different cluster groups are discussed. Finally, the detailed mechanism to predict local protein structure by CSVMs is explained.

9.2.1 Granular Computing

Basic principles of granular computing have been applied in many fields such as programming, artificial intelligence, interval computing, rough set theory, machine learning and database (Tang, Jin and Zhang, 2005; Yao, 2004). Granular computing can provide true and natural representation for natural, social and artificial systems.

Granular computing decomposes information in the form of some aggregates such as subsets, classes, and clusters of a universe and then solves the targeted problems in each granule (Yao, 2004). Granular construction and computing are two major tasks of granular computing (Yao, 2005). Granular computing conceptualizes the whole feature space at different granularities and switch among these granularities (Yao, 2004). With the principles of divide-and-conquer, granular computing breaks up the complex problems into smaller and computationally simpler problems and focuses on each small problem by omitting unnecessary and irrelevant information. As a result, granular computing can increase intelligence and flexibility of data mining algorithms.

In this study, SVM is utilized to learn the relationship between the protein sequence and its local 3D structure. Since different parts of the feature space may correspond to different 3D structures, building one SVM in the whole feature space may not be practical. It is more appropriate to separate the whole feature space into multiple subspaces with an effective granulation method and to model a SVM for each subspace. In this work, the k-means clustering algorithm is used as the granulation method. Since samples in the same subspace are closely related, SVM can be modeled more efficiently to capture inherent data distribution for these samples.

9.2.2 K-means Clustering Algorithm as the Granulation Method

Fuzzy sets, probabilistic sets, decision trees, clusters and association rules are some of granulation methods under the framework of granular computing (Yao, 2005). Since K-means clustering is computationally efficient for large data sets with both numeric and categorical attributes (Gupta, Rao, and Bhatnagar, 1999), improved K-means clustering algorithm introduced in Chapter 3 is chosen as the granulation method in our study. With the K-means

clustering algorithm, data samples with similar characteristics can be grouped together. As a result, the whole feature space is partitioned into subspaces intelligently and the complex data mining work is mapped into a series of computationally tractable simpler tasks. In order to compare the performance of the clustering system introduced previously and CSVMs, 800 initial clusters are selected for the improved K-means algorithm.

9.2.3 Generation of Sequence Segments by the Sliding Window Method

The sliding windows with eleven successive residues are generated from protein sequences. Each window represents one sequence segment of eleven continuous positions. Five hundred thousand sequence segments from the training set are produced by the sliding window method. The HSSP frequency profiles (Sander and Schneider, 1991) are chosen as the representation of sequence segments in this study. These sequence segments of eleven continuous positions are classified into different groups with the K-means algorithm.

9.2.4 Distance Score and Reliability Score of a Given Sequence Segment

The frequency profile for a given sequence segment is compared with the centroid of each cluster in order to calculate the distance score of the given sequence segment for each cluster. A smaller distance score shows that the frequency profile of the given sequence segment is closer to the centroid for the given cluster. The centroid of the related cluster is the average of all frequency profiles of sequence segments for this cluster. The following formula calculates the distance score of a given sequence segment for a specified cluster.

$$\text{Distance score} = \sum_{i=1}^{L} \sum_{j=1}^{N} |F_k(i, j) - F_c(i, j)| \qquad (79)$$

where L is the window size and N is 20. $F_k(i, j)$ is the value of frequency profile at row i and column j for the sequence segment k. $F_c(i, j)$ is the value of the matrix at row i and column j for the centroid of the cluster.

An average frequency profile summarizes how often amino acids occur in each position of a cluster. The following formula calculates the frequency of the amino acid of type R at the specified position of the average frequency profile for a sequence cluster:

$$f_R = \frac{Num_R}{total \quad number} \quad (80)$$

where Num_R is the number of amino acid of R in the specified position of the sequence cluster and $total \quad number$ is the total number of amino acids in the specified position of the sequence cluster.

The reliability score of a given sequence segment for a cluster is determined by the sum of the frequency of the matched amino acid in the corresponding position of the average frequency profile of a cluster. Higher reliability scores indicate that prediction results by this cluster is more dependable since the amino acids of the given sequence segment match more frequently occurring amino acids in the corresponding position of a cluster for structure conservation.

$$\text{Reliability score} = \sum_{i=1}^{L} F_c(i, j) \quad (81)$$

where $F_c(i, j)$ is the value of the matrix at row i and column j for the average frequency profile of the cluster. The value of j is determined by the type of amino acid in the specified position of sequence segment.

9.2.5 Cluster Membership Assignment for Each Sequence Segment

The distance score of each cluster for a given sequence segment is calculated in order to filter out some less significant clusters. If the difference of the clusters's distance score and the smallest distnace score is within 100, these clusters are selected. Other clusters are discarded since they are less significant. The cluster with the highest reliability score among these selected clusters finally functions to predict the structure of this sequence segment.

The distance score efficiently narrows down the list of possible clusters based on similarity of the frequency profile between the given sequence segment and the centroid of each cluster. The reliability score assesses how well amino acids of a given sequence segment match frequently occurring amino acids in the important positions of the average frequency profile for each cluster in order to conserve a particular local structure. Our experimental results show that the combination of the distance score and the reliability score can improve efficiency of the clustering membership function noticeably since the distance score and the reliability score carry independent biological information.

9.2.6 Support Vector Machine

Implementation of SVM has been explained in detail at Chapter 7. SVM (Vapnik, 1998) can handle a nonlinear classification efficiently by implicitly mapping input samples from the input feature space into another high dimensional feature space with the nonlinear kernel function. The classification boundary functions of SVM maximize the margin. In the machine learning theory, margin maximization corresponds to maximizing the generalization performance given a set of training data.

9.3 Clustering Support Vector Machines (CSVMs)

In this study, a new computational model called CSVMs (Clustering Support Vector Machines) is introduced. CSVMs creatively take advantages of granule computing and statistically learning theory in order to provide a new model for solving complex classification problems (Zhong et.al, Accepted for Publication).

9.3.1 Advantages of CSVMs

CSVMs are built from information granules, which are intelligently partitioned by clustering algorithms. Intelligent partitioning by clustering algorithms provides true and natural representations of inherent data distribution of the system. Because of data partitioning, a complex classification problem is converted into multiple smaller problems so that learning tasks for each CSVM are more specific and efficient (He et al., 2006). Each CSVM can concentrate on highly related samples in each feature subspace without being distracted by noisy data from other clusters. As a result, CSVMs can potentially improve the generalization capability for classification problems.

Since granulation by K-means clustering may introduce noise and irreverent information into each granule, the machine learning techniques are required to identify the strength of correspondence between frequency profiles and 3D local structure for each sequence segment belonging to the same information granule. After learning the relationship between frequency profile distribution and 3D local structures, CSVMs can filter out potentially unreliable prediction and can select potentially reliable prediction for each granule.

9.3.2 Training CSVMs for Each Cluster

Because our unpublished results reveal that the distribution patterns for frequency profiles in each cluster is quite different, the functionality and modeling of CSVMs is customized for each cluster belonging to different cluster groups. The definition of different cluster groups is introduced in the section 6.4.4**.** The CSVMs for clusters belonging to the bad and average cluster group are designed to identify sequence segments whose structure can be reliably predicted. As a result, the ratio of positive samples and negative samples is designed as 1 to 4 for the bad and average cluster group. The CSVMs for clusters belonging to the good cluster group are trained to filter out sequence segments whose structure cannot be reliably predicted. Therefore, the ratio of positive samples and negative samples is designed as 4 to 1 for the good cluster group.

The RBF kernel function is used for modeling each SVM. The RBF kernel parameters (j, γ, and C) are optimized by the grid search algorithm (Hsu, Chang, Lin, 2005). In each cluster, positive samples are defined as those samples whose structure deviation from the corresponding structure of this cluster is within a given threshold and negative samples are defined as those samples whose structure deviation from the corresponding structure of this cluster is above a given threshold. Frequency profiles of positive samples have the potential to be closely mapped to the given 3D local structure of the specified cluster and frequency profiles of negative samples may not correspond to the given 3D local structure of the specified cluster. Labeling sequence segments for each cluster as positive samples or negative samples provide training patterns for CSVMs to recognize the underlying association between frequency profiles and their 3D structure for each cluster.

9.3.3 Local Protein Structure Prediction by CSVMs

Local protein structure prediction by CSVMs is based on the prediction method from the clustering algorithm as introduced in Chapter 6. At first, the sequence segments whose structures to be predicted are assigned to a specific cluster in the cluster group by the clustering algorithm. Then CSVM trained for this specific cluster is used to identify how close the frequency profile of this sequence segment is nonlinearly correlated to the 3D local structure of this cluster. If the sequence segment is predicted as the positive sample by CSVM, the frequency profile of this segment has the potential to be closely mapped to 3D local structure for this cluster. Consequently, the 3D local structure of this cluster can be safely assigned to this sequence segment. The method to decide the 3D local structure of each cluster can be found in Chapter 6. If the sequence segment is predicted as the negative sample by CSVM, the frequency profile of this segment does not closely corresponds to the 3D local structure for this cluster. The structure of this segment cannot be reliably predicted by this cluster. This cluster is removed from the cluster group. The cluster membership function calculating distance scores and reliability scores is used to select the next cluster from the remaining clusters of the cluster group. The previous procedure will be repeated until one SVM modeled for the selected cluster predict the given sequence segment as positive. The complete prediction algorithm is shown in figure 15. Important knowledge about the correspondence between frequency profiles and the 3D local structure provided by CSVMs can provide the additional dependable metric of cluster membership assignment. Figure 15 shows our new CSVMs model. CSVMs are used to reclassify sequence segments, which are misclassified by the conventional clustering algorithms

Clustering Support Vector Machine Model

1. Granulating the whole sequence feature space into clusters by the K-means algorithm

WHILE (the training error is bigger than the threshold values)
{
Converting sequences into segments by the sliding window method
Assigning each segment to the specific cluster by membership functions
Updating the centroid and the frequency profile for each cluster
}

2. Training CSVM for each granule

Classifying clusters into different groups based on the training accuracy
FOR each cluster
{
Labeling each training sample as positive or negative respectively for different cluster groups
Modeling each CSVM for each cluster by optimizing RBF kernel parameters (j, γ, and C) with the grid search algorithm
}

3. Predicting protein structure by the CSVMs algorithm

While (there are clusters in the cluster group)
{
Allocating a given sequence segment to a cluster in the cluster group by membership functions
Predicting the property of the given sequence segment by CSVM modeled for the selected cluster
If (the given sequence segment is predicted as positive)
{
Assigning the corresponding structure of the selected cluster to this sequence segment
leave the loop
}
remove the selected cluster from the cluster group
}
randomly assigning a structure to the sequence segment

Figure 15 The CSVMs Model

9.4 Experimental Setup

9.4.1 Training Set and Independent Test Set

The training dataset used in our work includes 2000 protein sequences obtained from the Protein Sequence-Culling Server (PISCES) (Wang and Dunbrack, 2003). The training set is utilized to create sequence clusters and to model CSVM for each cluster. 200 protein sequences from the recent release of PISCES are included into the independent test set. The structures of

protein sequences in the training set and testing set are available from Protein Data Bank (PDB) (Berman et al. 2000). Any two sequences in the training set and the test set share less than 25% similarity.

9.4.2 Prediction Accuracy Calculation for Each Sequence Segment

Accuracy for structure prediction of sequence segments in terms of secondary structure accuracy, Distance Matrix Root Mean Square Deviation (dmRMSD) and Torsion angle RMSD (taRMSD) are calculated to evaluate the performance of the conventional clustering algorithm and our new computational model. The definition for average distance matrix and the representative torsion angle for a cluster was introduced in.

Q3 is one of the most commonly used performance measures in the protein secondary structure prediction. Q3 refers to the three-state overall percentage of correctly predicted residues. The following formula is used to calculate secondary structure accuracy (Hu et al. 2004):

$$Q3 = \frac{\sum_{i\in\{H,E,C\}} \#\ of\ residues\ correctly\ predicted_i}{\sum_{i\in\{H,E,C\}} \#\ of\ residues\ in\ class\ i} \tag{82}$$

The following formula is used to calculate dmRMSD (Zagrovic and Pande, 2004; Kolodny and Linial, 2004):

$$dmRMSD = \sqrt{\frac{\sum_{i=1}^{L}\sum_{j=i+1}^{L}\left(\alpha_{i\to j}^{s1} - \alpha_{i\to j}^{ADM}\right)^2}{M}} \tag{83}$$

$$M = \frac{(L \times L - L)}{2} \tag{84}$$

where $\alpha_{i\to j}^{ADM}$ is the distance between α-carbon atom i and α-carbon atom j in the average distance matrix of a cluster. M is the number of distances in the distance matrix in this formula.

The following formulas are used to calculate taRMSD:

$$taRMSD = \sqrt{\frac{\sum_{k=1}^{L}\left\{\left(\phi_{ki}-\phi_{kj}\right)^2 + \left(\psi_{ki}-\psi_{kj}\right)^2\right\}}{2L}} \quad (85)$$

where ϕ_{kj} is ϕ in the position k of the representative angle for a cluster and ψ_{kj} is ψ in the position k of the representative angle for a cluster. ϕ and ψ are defined in (Karp, 2002).

9.4.3 Classification of Clusters into Different Groups

During the prediction process, structures of sequence segments are first predicted by clusters with the high training accuracy. If the structures of sequence segments cannot be predicted by clusters with high training accuracy, clusters with the lower training accuracy will be used for structure prediction.

Training secondary structure accuracy for a given cluster is the average training accuracy of sequence segments in the training set predicated by this cluster. Training dmRMSD of a given cluster is the average training dmRMSD of sequence segments in the training set predicated by this cluster. Training taRMSD of a given cluster is similarly defined. Test secondary structure accuracy, test dmRMSD and test taRMSD is similarly defined for each cluster in the independent test set.

In the good cluster group, all clusters have training secondary structure accuracy greater than 80%, training dmRMSD less than 1 Å and training taRMSD less than 25 degree. The bad cluster group and the average cluster group are similarly defined. As a result, the good cluster group includes all the clusters with highest training accuracy. The bad cluster group includes clusters with poor training accuracy. The definition of the different cluster group is defined in the section 6.4.4.

9.4.4 Accuracy criteria for Each Cluster

In order to rigorously evaluate the prediction quality for these algorithms, we used two sets of accuracy criteria named accuracy criteria one and accuracy criteria two. Accuracy criteria one and accuracy criteria two considers secondary structure accuracy, dmRMSD and taRMSD simultaneously. Accuracy criteria two for one cluster is the percentage of sequence segments with secondary structure accuracy greater than 80%, dmRMSD less than 1 Å and taRMSD less than 25 degree in the test set for this cluster. Accuracy criteria two reflects the percentage of sequence segments with the most reliable structure prediction for one cluster. Accuracy criteria one is similarly defined. Accuracy criteria one reflects the percentage of sequence segments with acceptable level of structure prediction for one cluster. The defintion of accuracy criteria is defined in the section 6.4.5.

9.5 Experimental Results and Analysis

In this section, the accuracy, recall and precision of CSVMs for different cluster groups are shown. The local protein structure prediction performance of CSVMs and the conventional clustering algorithm is compared in order to demonstrate the advanced generalization capability of CSVMs.

9.5.1 Average Accuracy, Precision and Recall of CSVMs for Different Cluster Group

Figure 16 compares average accuracy, precision and recall of CSVMs for different cluster groups. Besides accuracy, precision and recall is also the important indicator for the generalization power of SVM. Only if values for accuracy, precision and recall are balanced, SVM can achieve satisfactory learning results. The equation 86 and 87 displays the formula for precision and recall. Figure 16 indicates that CSVM modeled for different cluster group obtains

good capability to discriminate between positive samples and negative samples. CSVMs for the bad cluster group are able to select frequency profiles of sequence segments whose structure can be reliably predicted. The recall value for CSVMs belonging to the good cluster group reaches 96%. This high value reveals that CSVMs did not misclassify frequency profiles of sequence segments whose structure can be accurately predicted. The precision value for CSVMs belonging to the good cluster group reaches 86%. The high precision value demonstrates that CSVMs belonging to the good cluster group obtain the capability to filter out the frequency profiles of sequence segments whose structure cannot be reliably predicted.

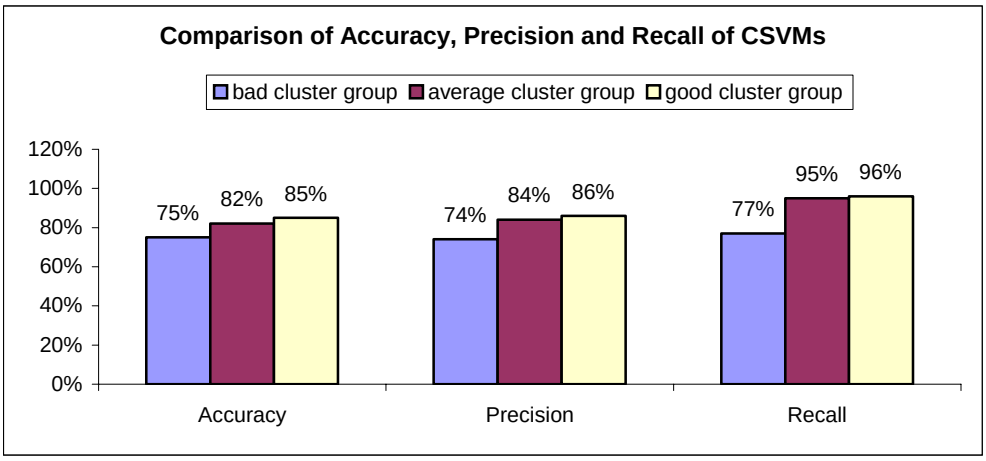

Figure 16 Comparison of Accuracy, Precision and Recall of CSVMs

$$precision = \frac{TP}{TP + FP} \qquad (86) \qquad recall = \frac{TP}{TP + FN} \qquad (87)$$

9.5.2 Comparison of Independent Prediction Accuracy for Different Cluster Groups in Terms of Three Metrics between the Clustering Algorithm and the CSVM Model

Figure 17 compares the secondary structure accuracy between the clustering system and the CSVMs model. Secondary structure accuracy for the bad cluster group increases by 8.32% when

the CSVM model is applied. Secondary structure accuracy for the average cluster group increases by 3.22% when the CSVM model is applied.

Figure 18 compares dmRMSD between the clustering system and the CSVMs model. The dmRMSD error for the bad cluster group reduces by 10.82% when the CSVM model is applied. The dmRMSD error for the average cluster group reduces by 6.90%. The dmRMSD error for the good cluster group reduces by 2.91% when the CSVM model is applied.

Figure 19 compares the taRMSD between clustering system and the CSVMs model. The taRMSD error for the bad cluster group reduces by 13.75% when the CSVM model is applied. The taRMSD error for the average cluster group reduces by 5.20% when the CSVM model is applied. The taRMSD error for the good cluster group reduces by 1.51% when the CSVM model is applied.

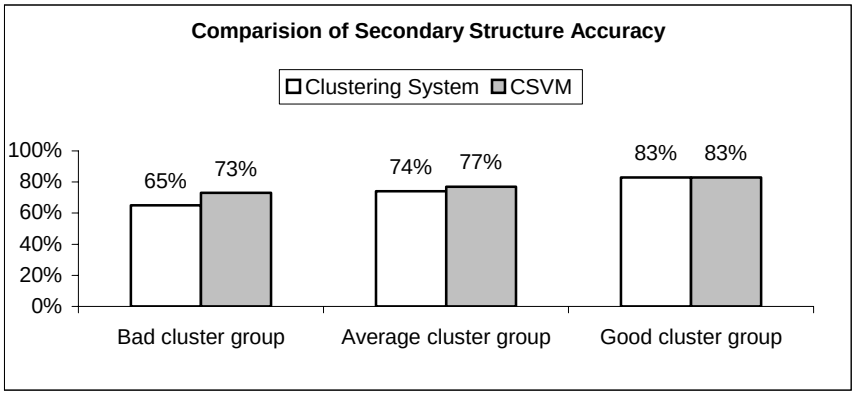

Figure 17 Comparison of Secondary Structure Accuracy between the Clustering System and CSVMs Model

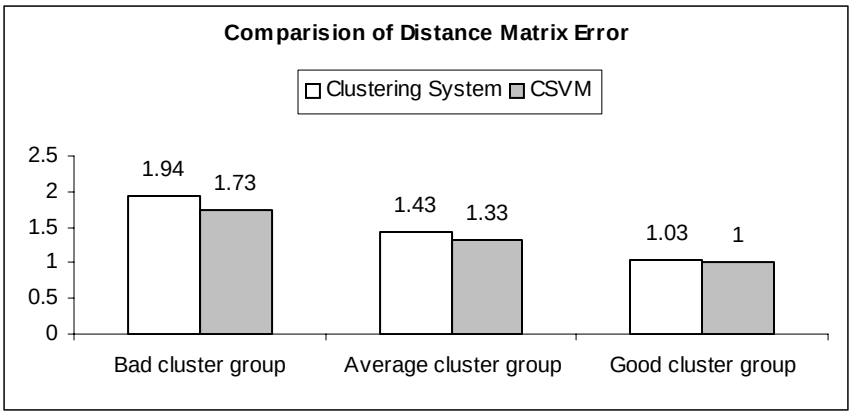

Figure 18. Comparison of dmRMSD between the Clustering System and CSVMs Model

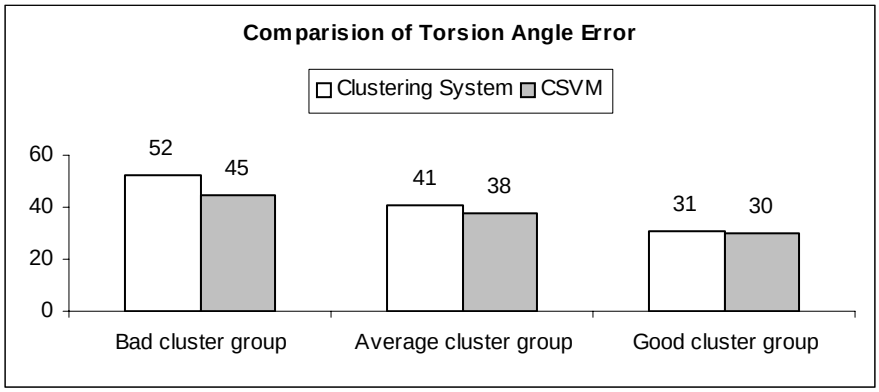

Figure 19. Comparison of taRMSD between the Clustering System and CSVMs Model

9.5.3 Comparison of Accuracy Criteria One and Accuracy Criteria Two between the Clustering System and the CSVMs Model

As described previously, accuracy criteria one and accuracy criteria two for local protein structure prediction have considered three evaluation metrics including secondary structure accuray, dmRMSD and taRMSD simultaneously. Since three metrics reflect the prediction accuracy in different perspectives, consideration of three metrics together will give the most rigorous evaluation for the quality of structure prediction. Accuracy criteria one reflects the percentage of sequence segments whose structural prediction is acceptable. Accuracy criteria two indicates the percentage of sequence segments whose structural prediction is the most reliable.

Figure 20 compares accuracy criteria one between the clustering system and the CSVMs model for different cluster groups. Figure 21 compares accuracy criteria two between the clustering system and the CSVMs model for different cluster groups.

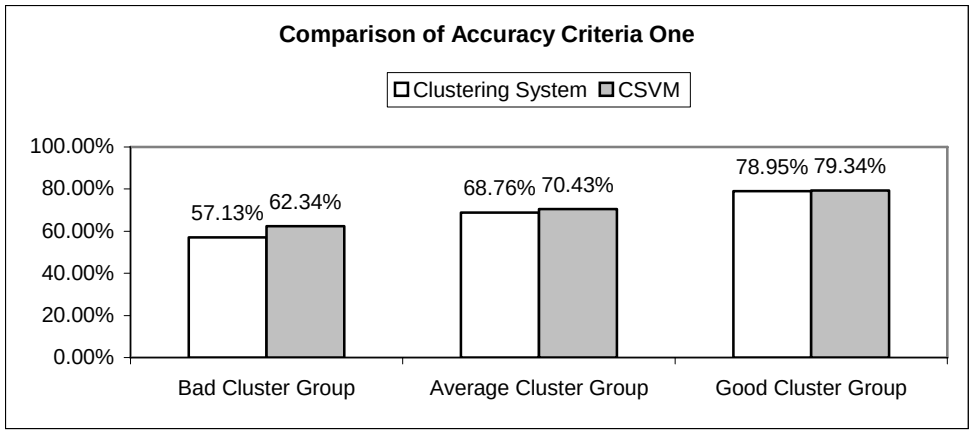

Figure 20. Comparison of Accuracy Criteria One between the Clustering System and The CSVMs Model for Different Cluster Groups

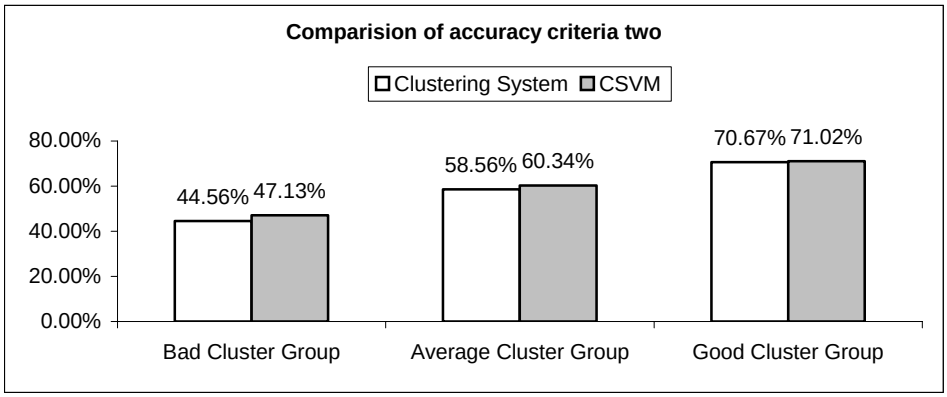

Figure 21. Comparison of Accuracy Criteria Two between the Clustering System and The CSVMs Model for Different Cluster Groups

Figure 20 and Figure 21 provide the evidences that the CSVMs model can improve the prediction accuracy under the most rigorous evaluation standard.

Average accuracy for CSVMs is over 80%, which indicates that the generalization power for CSVMs is strong enough to recognize the complicated pattern of frequency profiles for protein sequences. Compared with the clustering system, our experimental results show that accuracy for local structure prediction has been improved noticeably when the CSVMs model is applied.

With our experimental observation, the distribution patterns of frequency profiles for different cluster groups are not uniform. The distribution pattern of frequency profiles for the bad cluster group is diverse and the distribution pattern of frequency profiles for the good cluster group is compact. For different cluster groups, learning tasks for each CSVM are unique. Therefore, the customized CSVMs model can learn the sequence to structure relationship more specifically. Our experimental results indicate that modeling for each granule respectively can increase effectiveness and efficiency of CSVMs.

9.6 Summary

In previous works, the conventional clustering algorithm is used to capture the sequence-to-structure relationship. The clustering membership functions may not explore the nonlinear complex relationship effectively. To solve this problem, a new model called CSVMs (Clustering Support Vector Machines) is proposed. Each CSVM is customized to learn the unique frequency profile distribution in each cluster (Zhong et.al, Accepted for Publication). This strategy has increased the generalization capability for CSVMs. The superior performance of CSVMs provides a new approach to combine the granular computing and advanced statistical learning algorithms.

SVM is not efficient for very large datasets due to the high training time complexity. The special characteristics of CSVMs allow the training tasks for each CSVM to be parallelized. Parallel training process makes the data-mining task for very large datasets possible. The

satisfactory experimental results show that our new computational model opens a new approach for solving the complex classification problem in huge datasets.

Further improvement for the CSVMs model will be made in the future work (He et al., 2006). Currently, the greedy algorithm is utilized to select the next closest cluster if CSVM modeled for the assigned cluster predicts the sequence segment as negative. However, the greedy algorithm may not be optimal. The more effective fuzzy membership function need to be studied so that sequence segments can be assigned to a group of clusters with different membership weights.

Chapter 10 Conclusions and Future Work

Protein structure prediction is one of the open problems of computational biology today. Knowing the structure of a protein sequence enables us to probe the function of the protein, to perform drug design, and to construct novel proteins. Determination of protein structure can also provide important information for various researches such as mapping the functions of proteins in metabolic pathways for whole genomes. In this work, the performance of clustering system and CSVMs is compared. In order to explain the clustering system clearly, the improved K-means algorithm is introduced first. Then the relationship between sequence variation and structural variation for sequence clusters is explained. Based on this knowledge, the clustering system for local protein structure is discussed.

Several popular methods to develop sequence motifs are based on multiple sequence alignments. Multiple sequence alignment can reveal conserved regions for one family and cannot explore information across protein families. Furthermore, these popular methods depend on the existing knowledge about the biologically important regions or residues. As result, these methods for motif discovery are not automatic process. In contrast, our K-means clustering algorithm can universally conserved and elaborate sequence motifs across protein families. Furthermore, the clustering algorithm provides an automatic, unsupervised discovery process.

In order to overcome the problem of random selection, we propose the new greedy algorithm to select suitable initial points in order to allow the K-means algorithm to converge to a better local minimum (Zhong et.al, 2004a). The new greedy initialization method tries to choose suitable initial points so that final partitions can represent the underlying distribution of the data samples more consistently and accurately (Zhong et.al, 2004b). Each initial point is represented by one local sequence segment. In the new initialization method, structural similarity of

sequence clusters is evaluated after running the traditional clustering algorithm for several iterations during each run. Then the initial points producing clusters with high quality are selected. If the minimum evolutionary distance of these selected points is greater than the specified distance, these points is included into the initialization array. Satisfaction of the minimum evolutionary distance can guarantee that each newly selected point have the potential to fall into different natural clusters. This process will be repeated several times until 800 points is chosen.

Our experimental results shows that the average percentage of sequence segments belonging to clusters with high structural similarity steadily improves with increasing minimum evolutionary distances among initial points. This improved percentage results from decreased interferences among initial points when the evolutionary distances among initial points are increased. The increased average percentage and decreased standard deviation suggest that the improved K-means algorithm performs better and more consistently than the traditional algorithm because the improved K-means algorithm avoids outliers of clusters and keeps initial points as far as possible.

Analysis of related biochemical studies indicates that patterns obtained by the K-means algorithm may play vital roles in intramolecular interactions, which decide the structure and function of proteins. These patterns also influence intermolecular interaction, which affect how proteins communicate with other molecules. Furthermore, analysis of these sequence motifs provides important insight into the degrees to which changes in the primary sequence are tolerated. This knowledge can help us understand structurally conservative substitutions of 20 amino acids during the evolutionary process. The sequence motifs discovered in this study indicate conserved residues that are structurally and functionally important across protein

families because protein sequences used in this study share less than 25% sequence identities. Our sequence motifs may reflect general structural or functional characteristics shared by different protein families while sequence motifs from PROSITE, PRINTS, PFAM and BLOCKS represent structural or functional constraints specific to a particular protein family.

Testing the K-means clustering algorithm for sequence segments is a very slow and time consuming task because a large data set of thousands of amino acids and different algorithms have to be attempted for many times. However, the natural characteristics of the K-means algorithm allow itself to be easily parallelized because of its inherent data parallelism properties. In our project, two different parallelization methods using OpenMP and Pthread are used separately on the same K-means clustering algorithm and the performance for two parallelization methods are compared (Zhong et.al, Accepted for Publication). Hyper-Threading Technology enabled architecture is the test bed for both methods. Speedup for 16 Pthreads is 4.3 and speedup for 16 OpenMP threads is 4 in the 4 processors shared memory architecture. With the new parallel K-means algorithm, K-means clustering can be performed for multiple times in reasonable amount of time.

Bystroff and Baker have studied the relationship between sequence variation and structural. In their work, structural information is incorporated during the clustering process. As a result, final sequence clusters are contaminated by usage of structural information during the clustering process. Our implementation of the K-means clustering is significantly different from Bystroff's work (1998) because we only use recurrent clusters and do not include structural information in the clustering process so that the true relationship between protein structure variation and sequence variation for sequence clusters can be accurately reflected. Understanding this

relationship is very important to improve the quality of local sequence alignment and low homology protein folding.

The relative entropy is used to describe the extent to which the distribution of 20 amino acids in the specified position of the frequency profile is uniform. The relative entropy measures the difference between the amino acid equilibrium distribution of amino acids in the database and the distribution of amino acids in the specified position of frequency profiles. Larger entropy values reveal tight and increasingly imbalanced amino acid distribution in the specified position of the frequency profile and smaller entropy values represent increasingly uniform amino acid distribution in the specified position of the frequency profile. If the relative entropy in the specified position of the frequency profiles is greater than 0.2, this position is defined as the important position for frequency profiles. The number of important positions is used to assess sequence variation for sequence clusters. Increased number of important positions in the frequency profiles reflects more positions in the frequency profiles have highly disproportionate distribution of 20 amino acids. After the clustering process is completed, the structural variation of sequence-based clusters is evaluated by secondary structure similarity and dmRMSD_SC. Our results shows that the number of important positions for clusters with secondary structure similarity between 80% and 100% is greater than four. On the other hand, the majority of sequence clusters with secondary structural similarity between 50% and 60% have the important positions less than four. On average, the number of important positions for clusters with low structural variation is greater than the number of important positions for clusters with high structural variation.

The clustering system is used for local protein structure prediction. Cluster membership functions are important for correct assignment of sequence segments to the cluster. In this work,

the cluster membership functions calculate the distance score and reliability score. The distance score efficiently narrows down the list of possible clusters based on similarity of the frequency profile for the given sequence segment and the centroid of this cluster. The reliability score assesses how well the amino acids of a given sequence segment match key amino acids in the important positions in order to conserve a particular local structure. Our prediction results shows that the combination of the distance score and the reliability score can improve the prediction accuracy of the clustering system noticeably since the distance score and the reliability score carry very independent information. Our results show that the dmRMSD error for the average cluster group reduces by 26% compared to the bad cluster group. The dmRMSD error for the good cluster group reduces by 46% compared to the bad cluster group. Accuracy of the good group cluster has improved by 17% compared to the bad cluster group in terms of accuracy criteria one. All our experimental results indicate that clusters with high quality provide the reliable prediction results and clusters with average quality produces high quality results. Special cautions need be taken against prediction results by the bad cluster group.

In our clustering system for local protein structure prediction, the K-means clustering algorithm is essential to understand how protein sequences correspond to local 3D protein structures. To the best of our knowledge, the sequence-to-structure relationship is nonlinear. However, the conventional clustering algorithms assume that the distance between data points can be calculated with exact precision. When this distance function is not well characterized, the clustering algorithm may not capture this nonlinear the sequence-to-structure relationship effectively. SVM can handle nonlinear relationship efficiently by implicitly transforming the input space into another higher dimensional space. However, SVM is not favorable for huge datasets training. In our test, training of the half million samples is not completed after one

month on the "poweredge6600 server" with four processors from Dell®. According to Hwanjo Yu, Jiong Yang, and Jiawei Han (2003), it would take years to train SVMs on a data set containing one million records. In order to solve the problem of training the large sample, the Clustering Support Vector Machines is proposed.

Fuzzy sets, probabilistic sets, decision trees, clusters and association rules are some of granulation methods under the framework of granular computing (Yao, 2005). Since K-means clustering is computationally efficient for large data sets with both numeric and categorical attributes (Gupta, Rao, and Bhatnagar, 1999), the improved K-means clustering algorithm introduced in Chapter 3 is chosen as the granulation method in our study.

CSVMs are built from information granules. These information granules are intelligently partitioned by clustering algorithms. Intelligent partitioning by clustering algorithms can make the data mining task easier by gaining better understanding the true and natural representations of inherent data distribution of the system. Because of data partitioning, a complex classification problem is converted into multiple smaller problems so that learning tasks for each CSVM are more specific and efficient (He et al., 2006). Each CSVM can concentrate on highly related samples in each feature subspace without being distracted by noisy data from other clusters. As a result, CSVMs can potentially improve the generalization capability for classification problems. Besides local structure prediction problem, CSVMs can be applied to the structured data in general. For structured data, several underlying sample subspaces have the unique data distribution pattern. It is inappropriate to build one SVM over the whole sample space. It is much better to divide the whole sample space into multiple sample subspaces and to build the SVM over each sample space. As a result, the generalization capability of the SVMs can be improved. Our experimental results indicate that the average accuracy for CSVM almost reaches 80%. This

high accuracy value shows that CSVM has already obtained the strong capability to identify the complex pattern of the sequence-to-structure relationship for each cluster. The dmRMSD error for the bad cluster group reduces by 10.82% when the CSVM model is applied. The dmRMSD error for the average cluster group reduces by 6.90%. The dmRMSD error for the good cluster group reduces by 2.91% when the CSVM model is applied. Compared with the clustering system, our experimental results show that accuracy for local structure prediction has been improved noticeably when the CSVMs model is applied.

Three metrics including the decision value from SVM, the distance score and the reliability score are used to give the final cluster membership assignment. As introduced previously, our cluster membership functions use the greedy algorithm. The greedy algorithm may not be optimal. As a result, the accuracy improvement for the clusters belonging to the good cluster group and the average cluster group is not significant. In order to improve the accuracy for protein structure prediction, two new cluster membership functions are proposed for the future work. In the first cluster membership function, the sequence segment is assigned to the cluster with the maximum SVM decision value. The second cluster membership will be based on the information fusion from the decision value from SVM, the distance score and the reliability score.

The different cluster has diverse distribution pattern of the frequency profiles. The customized kernel function can control the upper bound of testing error more effectively. In the next step, we need develop the function to estimate density of each cluster effectively. Based on accurate density estimation, the effective kernel function can be derived. The customized kernel function can further improve the prediction accuracy.

As introduced previously, important positions play key roles in determining the sequence and structural variation for sequence clusters. The features from unimportant positions are used during the training process of SVM. The information from unimportant positions may introduce the noisy and irrelevant information to increase errors for SVM. In order to increase the generalization capability of SVM, I propose only using features from important positions and discarding the information related to unimportant positions. The new cluster membership function, kernel selection and feature selection can be very effective to improve the accuracy of CSVMs.

In the next step, we need make further analysis about the relationship and interactions among these sequence clusters. Since the CSVMs are trained specifically for each information granule, the CSVMs can be easily parallized to address the problem of large dataset training. In the next step, the comparative study of parallel SVM and CSVMs need be carried out in order to show the advantage of CSVMs.

After implementing the parallel K-means algorithm, we need test the performance of the improved K-means algorithm with the minimum distance of 2000 and the minimum distance of 3000. According to the experimental results, the performance of the improved K-means algorithm may be improved further.

Bibliography

S. F. Altschul, W. Gish, W. Miller, E.W. Myers, D. J. Lipman, "Basic local alignment search tool," J. Mol. Biology, vol. 215, pp. 403, 1990.

S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller and D. J. Lipman, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," Nucleic Acids Res., vol. 25, no. 17, pp. 3389-3402, 1997.

T. K. Attwood, M. Blythe, D. R. Flower, A. Gaulton, J. E. Mabey, N. Maudling, L. McGregor, A. Mitchell, G. Moulton, K. Paine, and P. Scordis, "PRINTS and PRINTS-S shed light on protein ancestry," Nucleic Acids Res., vol. 30, no. 1, pp. 239-241, 2002.

K. Alexander, "Probability inequalities for empirical process and law of interated logarithm," Ann. Probab. vol. 4, pp.1041-1067, 1984.

M. A. Aizerman, E. M. Braverman and L. I. Rozonoer, "The problem of pattern recognition learning and the method of potential functions," Autom. Remote Control, vol. 25, pp.821-837, 1964.

R. Aurora and G. D. Rose, "Helix capping," Protein Sci., vol. 7, no. 1, pp. 21-38, 1998.

J. L. Balcazar, Y. Dai and O.Watanabe, "Provably Fast Training Algorithms for Support Vector Machines," in Proc.of the 1stIEEE International Conference on Data mining, IEEE Computer Society, pp.43-50, 2001.

N. Barakat, J. Diedrich, "Learning-based rule-extraction from support vector machine," in Proc. Of NCEI, 2004.

J. M. Berg, J. L. Tymoczko and L. Stryer, Biochemistry, pp. 53-70, fifth edition, W.H. Freeman, New York, 2002.

N. Bebiano, R. Lemos and J. D. Providencia, "Inequalities for quantum relative entropy," Linear Algebra and its Application, vol. 401, pp. 159-172, 2005.

H. M. Berman, J. Westbrook…and P. E. Bourne, "The protein data bank," Nucleic Acids Research, vol. 28, pp. 235-242, 2000.

J. U. Bowie, R. Luthy and D. Eisenberg, " A method to identify protein sequences that fold into a known three-dimentional structure," Science, vol. 253, pp. 164, 1991.

R. Bonneau, J. Tsai, I. Ruczinski, C. rohl, C. E. Strauss and D. Baker, "ROSETTA in CASP4: Progress in Ab Initio protein structure prediction," Proteins: Structure, Funcions and Genetics, vol. 119, 2001.

D. R. Butenhof, “Programming with POSIX Threads,” Addison-Wesley Professional Computing Series, 1997.

C.Bystroff and D. Baker, “Prediction of local structure in proteins using a library of sequence-structure motifs.” J. Mol. Biol., vol. 281, pp. 565-577, 1998.

C. Bystroff, V. Thorsson and D. Baker, “HMMSTR: a Hidden Markov Model for Local Sequence-Structure Correlations in Proteins,” J. Mol. Biol., vol. 281, pp. 173-90, 2000.

C. C Chang and C. J. Lin, “Training nu-support vector classifiers: Theory and algorithms,” Neural Computations, vol. 13, pp. 2119-2147, 2001.

R. Chandra, L. Dagum, D. Kohr, D. Maydan, J. McDonald, and R.Menon, “Parallel Programming in OpenMP,” Morgan Kaufmann Publishers, 2000.

Y. Chen, C. T. Mant and R. S. Hodges, “Determination of stereochemistry stability coefficients of amino acid side-chains in an amphipathic a-helix,” J. Peptide Research, vol.59, pp. 18-33, 2002.

S. Chung and S. Subbiah, “ A structural explanation for the twilight zone of protein sequence homology,” Structure, vol. 4, pp. 1123, 1996.

N. Cristianini, J. Shawe-Taylor, “An Introduction to Support Vector Machines and other Kernel-based Learning Methods,” Cambridge University Press, Cambridge, UK, 2000.

W. F. DeGrado, “Design of peptides and proteins,” Adv. Prot. Chem., vol. 39, pp. 51-124, 1988.

L. Devroye and L. Gyorfi, A Probabilistic Theory of Pattern Recognition, Springer, New York, 1996.

R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, “Biological sequence analysis: probabilistic models of protein and nucleic acids,” Cambridge University Press, 1998.

S. Eggers, J. Emer, H. Levy, J. Lo, R. Stamm, and D. Tullsen, “Simultaneous Multithreading: A Platform for Next-generation Processors,” IEEE Micro, pp. 12-18, 1997.

J. Finer-Moore and R. M Stroud, “Amphipathic analysis and possible formation of the ion channel in an acetocholine receptor,” Proc. National Academy of Sciences, USA, vol. 81, no. 1, pp. 155-159, 1984.

D. Frishman and P. Args, “Knowledge-based protein secondary structure assignment,” Proteins: Structure, Function, and Genetics, vol. 23, pp. 566-579, 1995.

S. K. Gupta, K. S. Rao, and V. Bhatnagar, “K-means clustering algorithm for categorical attributes,” Data Warehousing and Knowledge Discovery DaWaK-99, pp. 203-208, Italy, 1999.

R. W. Harrsion, C. Devjani and I. T. Weber, “Analysis of six protein structures predicted by comparative modeling techniques,” Protein, vol. 23, pp. 463-71, 1995.

K. F. Han and D. Baker, “Recurring local sequence motifs in proteins,” J. Mol. Biol., vol. 251, no. 1, pp. 176-187, 1995.

K. F. Han and D. Baker, “Global properties of the mapping between local amino acid sequence and local structure in proteins,” Proc. Natl. Acad. Sci. USA, vol. 93, no. 12, pp. 5814-5818, 1996.

C. W. Hsu, C.C. Chang, C.J. Lin, "A practical guide to support vector classification," Available at http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf, 2005.

E. G. Hutchinson and J. M. Thornton, “A revised set of potentials for ß-turn formation in proteins,” Protein Sci., vol. 3, no. 12, pp. 2207-2216, 1994.

S. Henikoff, J. G. Henikoff and S. Pietrokovski, “Blocks+: a non-redundant database of protein alignment blocks derived from multiple compilations,” Bioinformatics, vol. 15, no. 6, pp. 471-479, 1999.

J. He, W. Zhong, R. Harrison, P. C. Tai and Y. Pan, “Clustering Support Vector Machines and Its Application to Local Protein Tertiary Structure Prediction," Proceedings of International Workshop on Bioinformatics Research and Applications (IWBRA2006), 2006.

U. Hobohm, M. Scharf, R. Schneider and C. Sander, “Selection of representative protein data sets,” Protein Sci., vol.1, no. 3, pp. 409-417, 1992.

N. Hulo, C. J. A. Sigrist, V. Le Saux, P. S. Langendijk-Genevaux, L. Bordoli, A. Gattiker, E. De Castro, P. Bucher, and A. Bairoch, “Recent improvements to the PROSITE database,” Nucleic Acids Res., vol. 32, pp. 134-137, 2004.

H. Hu, Y. Pan, R. Harrsion and P.C. Tai , “Improved protein secondary structure prediction using support vector machine with a new encoding scheme and ad-vanced tertiary classifier,” IEEE Transactions on NanoBioscience, vol. 2, pp. 265-271, 2004.

D. T. Jones, W. R. Taylor and J. M. Thornton, “ A new approach to protein fold recognition. Nature, vol. 358, pp. 86, 1992.

A. K. Jain, M. N. Murty and P. J. Flynn, “Data clustering: a review,” ACM Computing Surveys, vol. 31, no. 3, pp. 264-323, 1999.

A. Juan and E. Vidal, “Comparison of four initialization techniques for the K-Medians Clustering Algorithm,” 8th International Workshop on Structural and Syntactic Pattern Recognition, 3rd International Workshop on Statistical Techniques in Pattern Recognition, vol. 1876, pp. 842-852, 2000.

C. Johansson and A. Lansner, "A Parallel Implementation of a Bayesian Neural Network with Hypercolumns," Technical Report, TRITA-NA-P0121, SANS-Nada-KTH, 2001.

I. Jonassen, I. Eidhammer, S. H. Grindhaug and W. R. Taylor, "Searching the protein structure databank with weak sequence patterns and structural constraints," J. Mol. Biol, vol. 304, pp. 599-619, 2000.

G. Karp, Cell and molecular biology (Concepts and Experiments), pp. 52-65, third Edition, John Wiley & Sons Inc, 2002.

W. Kauzmann, "Some factors in the interpretation of protein denaturation," Adv. Protein Chem., vol. 14, pp. 1-63, 1959.

M. Gerstein and M. Levitt, "Comprehensive assessment of automatic alignment against a manual standars: the SCOP classification of proteins," Protein Sci., vol. 7, pp. 445, 1998.

V. Guralnik and G. Karypis, "A scalable algorithm for clustering protein sequences," Workshop on Data Mining in Bioinformatics (BIOKDD), pp. 73-80, 2001.

P. Koehl and M. Levitt, " Improved recognition of native-like proteins tructures using a family of designed sequences," Proceeding of National Acacdemy Science, vol. 99, pp. 691, 2002.

A. Kasuya, J.M. Thornton, “Three-dimensional structure analysis of PROSITE patterns,” J. Mol. Biol., vol. 286, pp. 1673-91, 1999.

R. Kolodny and N. Linial, “Approximate protein structural alignment in polynomial time.” Proc Natl. Acad. Sci., vol. 101, pp. 12201-12206, 2004.

E. T. Kaiser and F. J. Kézdy, “Secondary structures of proteins and peptides in amphiphilic environments. (A review),” Proc. Natl. Acad. Sci., vol. 80, no. 4, pp. 1137-1143, 1983.

W. Kabsch and C. Sander, “Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features,” Biopolymers, vol. 22, pp. 2577-2637, 1983.

R. Kolodny and N. Linial, “Approximate protein structural alignment in polynomial time,” Proc Natl. Acad. Sci., Vol. 101, pp. 12201-12206, 2004.

J. Kyte and R. F. Doolitle, “A simple method for displaying the hydropathic character of a protein,” J. Mol.Biol., no. 157, pp. 105-132, 1982.

S. Lifson and C. Sander, “Antiparallel and parallel beta-strands differ in amino acid residue preferences,” Nature, vol. 282, no. 5734, pp. 109-111, 1979.

C. T. Mant, N. E. Zhou and R. S. Hodges, “The Role of Amphipathic Helices in Stabilizing Peptide and Protein Structure,” The Amphipathic Helix (Epand, R.M., ed.), CRC Press, Boca Raton, FL, USA, 1993.

D. T. Marr, F. Binns, D. L. Hill, G. Hinton, D. A. Koufaty, J. A. Miller, M. Upton, "Hyper-Threading Technology Architecture and Microarchitecture," Intel Technology Journal, 2002.

A. G. Murzin, S. E. Brenner, T. Hubbard and C. Chothia, “SCOP: a structural classification of proteins database for the investigation of sequences and structure,” J. Mol. Biol., vol. 247, pp. 536, 1995.

T. Noguchi, H. Matsuda, and Y. Akiyama, “PDB-REPRDB: a database of representative protein chains from the Protein Data Bank (PDB),” Nucleic Acids Res., vol. 29, no. 1, pp. 219-220, 2001.

A.B.J. Novikoff, “On convergence proofs on perceptrons,” Proceedings of the Symposium on the Mathematical Theory of Automata, Vol. XII, Polytechnic Institute of Brooklyn, pp. 615-622, 1962.

E.Osuna, R.Freund, and F.Girosi. An improved training algorithm for support vector machines In Proc. Of IEEE Workshop on Neural Networks for Signal Processing, pp. 276-285.1997.

J.Platt, “Fast training of support vector machines using sequential minimal optimization,” advances in Kerenel Methods-Support Vector Learning, pp. 185-208, 1999.

J. M Peña., J. A Lozano and P. Larrañaga, “An empirical comparison of four initialization methods for the K-Means algorithm,” Pattern Recognition Letters, vol. 20, no. 10, pp. 1027-1040, 1999.

D. Pavlov, J. Mao, J. B. Dom, “Scaling-up support vector machines using boosting algorithm Proceedings,” Proceeding of 15th International Conference on Publication Date, Vol. 2, pp. 219-222, 2000.

M. J. D. Powell, “The theory of radius basis functions approximation in 1990,” Advances in Numerical Analysis volum II: Wavelets, Subdivision Algorithms and Radial Basis Funcions, W.A. Light, ed., Oxford University, pp. 105-210, 1992

P. L. Privalov, “Thermodynamics of protein folding,” J. Chem. Thermodyn., vol. 29, pp. 447-474, 1997.

A. Rahman and A. Y. Zomaya A.Y. “An overview of protein-folding techniques: issues and perspectives,” IJBRA, vol. 1, pp. 121 – 143, 2005.

A. Rahman and A. Y. Zomaya, “An overview of protein-folding techniques: issues and perspectives,” International Journal of Bioinformatics Research and Applications, vol. 1, pp. 121 – 143, 2005,

F. M. Richards and C. E. Kundrot, “Identification of structural motifs from protein coordinate data: Secondary structure and first-level supersecondary structure,” Proteins: Struct. Funct. Genet., vol. 3, pp. 71-84, 1988.

F. Rosenblatt. Principles of Neurodynamics: Perceptron and Theory of Brain Mechanisms, Spartan Books, Washington, DC, 1962.

B. Rost, “ Protien structure sustain evolutionary drift,” Fold Des., vol. 2, pp. 519, 1997.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation, Parallel Distributed Processing: Explorations in the Macrostructure of Cognition, Vol. I, Bradford Books, Cambridge, MA, pp. 318-362, 1986.

S. Salas, E. Hille, G. Etgen, Calculus One and Several Variables, pp. 729, Ninth Edition, John Wiley & Sons Inc, 2003.

M. Schiffer and A. B. Edmundson, “Use of helical wheels to represent the structures of proteins and to identify segments with helical potential,” Biophysical J., vol. 7, no. 2, pp. 121-135, 1967.

C. Sander and R. Schneider, “Database of homology-derived protein structures and the structural meaning of sequence alignment,” Proteins: Struct. Funct. Genet., vol. 9, no. 1, pp. 56-68, 1991.

B. Schoelkopf, K. Tsuda and J.P. Vert, “Kernel Methods in Computational Biology,” MIT Press, pp.71-92, 2004.

G.Scholhn and D. Cohn, “Less or more: Active learning with support vector machines,” Proceedings of 17th Int. Conf. Machine Learning, 2000.

J. P. Segrest, H. De Loof, J. G. Dohlman, C. G. Brouilette and G. M. Anantharamaiah, “Amphipathic helix motif: classes and properties,” Proteins: Struct. Funct. Genet., vol. 8, no. 2, pp. 103-117, 1990.

J. M. Sauder, W. Arthur, and R.L. Dunbrack, “Large-scale comparision of proteins equence alignment algorithms with structure alignment,” Protein: Struct., Func., and Genetics, vol 40, pp. 6, 2000.

Y. Sun, Q. Zhu and Z. Chen, “An iterative initial-points refinement algorithm for categorical data clustering,” Pattern Recognition Letters, vol. 23, no. 7, pp. 875-884, 2002.

J. Selbig and P. Argos, “Relationships between protein sequence and structure patterns based on residue contacts proteins,” Proteins: Struct. Funct. Genet., vol. 31, pp. 172-185, 1998.

K. T. Simons, C. Kooperberg, E. Huan and D. Baker, “ Assembly of protein tertiary structures from framents with similar local sequences using simulated annealing and Bayesian scoring functions,” J. Mol. Biol., vol. 268, pp. 209, 1997.

E. L. L. Sonnhammer, S. R. Eddy, E. Birney, A. Bateman and R. Durbin, “Pfam: multiple sequence alignments and HMM-profiles of protein domains,” Nucleic Acids Research, vol. 26, no. 1, pp. 320-322, 1998.

Y.C. Tang, B. Jin and Y.-Q. Zhang, “Granular Support Vector Machines with Association Rules Mining for Protein Homology Prediction,” Special Issue on Computational Intelligence Techniques in Bioinformatics, Artificial Intelligence in Medicine, vol. 35, no. 1-2, pp. 121-134, 2005.

R. K. Thulasiram, R. M. Rahman and P. Thulasiraman, “Neural Network Training Algorithms on Parallel Architectures for Finance Applications,” ICPP Workshops, pp. 236-243, 2003.

X. Tian, A. Bik, M. Girkar, P. Grey, H. Saito, and E. Su, “Intel OpenMP C++/Fortran Compiler for Hyper-Threading Technology: Implementation and Performance,” Intel Technology Journal, 6 (1), 2002.

V.N. Vapnik and A. Y. Chervonenkis, “the necessary and sufficient conditions for consistency of the method of empirical risk minimization,” Pattern Recogn. Image Anal, vol. 1, pp. 285-305, 1991.

V. Vapnik, Statistical Learning Theory. John Wiley&Sons, Inc., New York, 1998.

S. A. Vavasis, “Nonlinear Optimization: Complexity Issues,” New York: Oxford Science, 1991.

G. Wang and R. L. Dunbrack, Jr., “PISCES: a protein sequence-culling server,” Bioinformatics, vol. 19, no. 12, pp.1589-1591, 2003.

Y. Xu, D. Xu, O.H. Crawford and J. R. Einstein, “ A computational method for NMR-constrained protein threading,” vol. 7, pp. 449, 2000.

Y.Y. Yao, “Granular Computing,” Computer Science (Ji Suan Ji Ke Xue), Proceedings of The 4th Chinese National Conference on Rough Sets and Soft Computing, vol. 31, pp.1-5, 2004.

Y.Y. Yao, “Perspectives of Granular Computing,” Proceedings of 2005 IEEE International Conference on Granular Computing, vol. 1, pp. 85-90, 2005.

H. Yu, J. Yang, and J.W. Han, “Classifying Large Data sets Using SVMs with Hierarchical Clusters,” Proceedings Of the 9th ACM SIGKDD, pp. 306-315, 2003.

W. Zhong, G. Altun, R. Harrison, P. C. Tai, and Y. Pan, “Discovery of Local Protein Sequence Motifs Using Improved K-means Clustering Technique,” Proceedings of International Conference on Bioinformatics and Its Applications (ICBA2004), pp.297-306, 2004a.

W. Zhong, G. Altun, R. Harrison, P. C. Tai, and Y. Pan, “Factoring Tertiary Classification into Binary Classification Improves Neural Network for Protein Secondary Structure Prediction,” Proceedings of IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB2004), pp. 175-181, 2004b.

W. Zhong, G. Altun, R. Harrison, P. C. Tai, and Y. Pan, “Parallel Protein Secondary Structure Prediction Based on Neural Networks,” Proceedings of 26th Annual International Conference of the IEEE Engineering in Medicine and Biology (EMBC2004), pp.2968-2971, 2004c.

W. Zhong, G. Altun, R. Harrison, P. C. Tai, and Y. Pan, “Mining Relationship between Structural Homology and Frequency Profile for Structure Clusters,” Poster Paper of the Ninth Annual International Conference on Research in Computational Molecular Biology (RECOMB2005), pp. 397-398, 2005a.

W. Zhong, G. Altun, R. Harrison, P. C. Tai, and Y. Pan, “Mining Protein Sequence Motifs Representing Common 3D Structures,” Poster Paper of 2005 IEEE Computational Systems Bioinformatics (CSB2005), pp. 272-274, 2005b.

W. Zhong, G. Altun, R. Harrison, P. C. Tai, and Y. Pan, "Improved K-means Clustering Algorithm for Exploring Local Protein Sequence Motifs Representing Common Structural Property," IEEE Transactions on NanoBioscience, vol. 4, no. 3, pp. 255-265, September 2005c.

W. Zhong, J. He, R. Harrison, P. C. Tai, and Y. Pan, “Clustering Support Vector Machines for Local Protein Structure Prediction,” Expert Systems with Applications: An International Journal, accepted for publication.

W. Zhong, G. Altun, R. Harrison, P. C. Tai, and Y. Pan, "Parallel Protein Secondary Structure Prediction Schemes Using Pthread and OpenMP over Hyper-Threading Technology," Journal of Supercomputing, accepted for publication.

B. Zagrovic and V. S. Pande, “How does averaging affect protein structure comparison on the ensemble level?” Biophysical Journal, vol. 87, pp. 2240-2246, 2004.

L. A. Zadeh, “Towards a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic,” Fuzzy Sets and Systems, vol. 19, pp. 111-127, 1997.

J.M. Zimmerman, N. Eliezer and R. Simha, “The characterization of amino acid sequences in proteins by statistical methods,” J. Theor. Biol., vol. 21, pp. 170-201, 1968.