5-3-2007

Protein Secondary Structure Prediction Using Support Vector Machines, Nueral Networks and Genetic Algorithms

Anjum B. Reyaz-Ahmed

PROTEIN SECONDARY STRUCTURE PREDICTION USING SUPPORT VECTOR MACHINES, NUERAL NETWORKS AND GENETIC ALGORITHMS

by

ANJUM B REYAZ-AHMED

Under the Direction of Yanqing Zhang

ABSTRACT

Bioinformatics techniques to protein secondary structure prediction mostly depend on the information available in amino acid sequence. Support vector machines (SVM) have shown strong generalization ability in a number of application areas, including protein structure prediction. In this study, a new sliding window scheme is introduced with multiple windows to form the protein data for training and testing SVM. Orthogonal encoding scheme coupled with BLOSUM62 matrix is used to make the prediction. First the prediction of binary classifiers using multiple windows is compared with single window scheme, the results shows single window not to be good in all cases. Two new classifiers are introduced for effective tertiary classification. This new classifiers use neural networks and genetic algorithms to optimize the accuracy of the tertiary classifier. The accuracy level of the new architectures are determined and compared with other studies. The tertiary architecture is better than most available techniques.

INDEX WORDS: Binary classifier, BLOSUM62, encoding scheme, orthogonal profile, support vector machine (SVM), tertiary classifier.

PROTEIN SECONDARY STRUCTURE PREDICTION USING SUPPORT VECTOR MACHINES, NUERAL NETWORKS AND GENETIC ALGORITHMS

by

ANJUM B REYAZ-AHMED

Under the Direction of Yanqing Zhang

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

Master of Science

in the College of Arts and Sciences

Georgia State University

2007

PROTEIN SECONDARY STRUCTURE PREDICTION USING SUPPORT VECTOR MACHINES, NUERAL NETWORKS AND GENETIC ALGORITHMS

by

ANJUM B REYAZ-AHMED

Under the Direction of Yanqing Zhang

ACKNOWLEDGEMENTS

I wish to take this opportunity to thank many people without whom this thesis would not have been accomplished. First and foremost, I would like to thank my thesis advisor, Dr. Yanqing Zhang. I was able to achieve this task, with his help, guidance, encouragement, and the time that he has spent on directing my thesis.

I also wish to thank my thesis committee members, Dr. Saeid Belkasim and Dr. Yingshu Li, for taking time to evaluate my simulation results and to review my thesis document. I would like to express my gratitude to Dr. Hyunsoo Kim from Georgia Tech, who helped me understand the PSSM profiling.

I would like to thank my fellow department members and all my friends, who patiently listen to all my doubts and queries and helped me in many ways.

Last but not least, I wish to express my gratitude to my parents and my brother who have pushed me this far. I most certainly should thank my husband Ashraf, who supported me and encouraged me even when I kept bugging him.

Declaration

I hereby declare that, except where otherwise indicated, this document is entirely my own work and has not been submitted in whole or in part to any other university.

Signed: ....Date: ....

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATION

BLOSUM	Block Substitution Matrix
DAG	Directed Acyclic Graph
DSSP	Database of Secondary Structure in Proteins
ERM	Empirical Risk Minimization
OSH	Optimal Separating Hyperplane
PSSM	Position Specific Scoring Matrix
RBF	Radial Basis Function
SOV	Segment Overlap Measure
SRM	Structural Risk Minimization
SVM	Support Vector Machines

1. INTRODUCTION

Protein secondary structure is closely related to the protein tertiary structure, which determines the characteristically behavior of the proteins. Many researches have been done over the decades to study and predict the protein structure. Till date, the total number of experimentally determined structures is less than twenty thousand (Protein Data Bank) whereas there are over a million known protein sequences. It is therefore becoming increasingly important to predict protein structure from its amino acid sequence, using insight obtained from already known structures.

Neural networks or the support vector machines are the generally adopted techniques to predict protein secondary structure. The SVM method is a comparatively new learning system which has mostly been used in pattern recognition problems. This machines uses hypothesis space of linear functions in a high-dimensional feature space, and it is trained with a learning algorithm based on optimization theory. To compare the results of this study with previous results RS126 data set is used. RS 126 data set [1] is proposed by Rost and Sander. Among neural networks Chandonia and Karplus [2] introduced a novel method for processing and decoding the protein sequence with NNs by using large training data set such as 681 non homologous proteins. And with the use of jury method, this scheme recorded 74.8% accuracy. Some of the recent studies adopting this SVM learning machine for secondary structure prediction are the one which used frequent profiles with evolutionary information as an encoding schemes for SVM [3], the one which adopted two layers of SVM, with a weighted cost function [4] and the one which applied PSI-BLAST PSSM profiles [5] as an input vector and sliding window scheme with SVM_Representative architecture [6]. All then show good prediction accuracies over 70%.

Support Vector Machine is learning system that uses a high dimensional feature space, trained with a learning algorithm from optimization theory. Since SVM has many advantageous features including effective avoidance of over-fitting, the ability to manage large feature spaces, and information condensing of the given data, it has been gradually applied to pattern classification problem in biology.[7]

In this study, the single sliding window scheme used to form the data for binary classifier is challenged with multiple windows scheme. Both the binary classifiers show more or less the same accuracies. In some cases single window is better and in some multiple. Since the optimal window size is already determined by Hu's method [6] the same is used here.

Then two novel tertiary architectures are introduced. In both the architectures the results have the better accuracy when compared to the method proposed by Hu [6]. In Hu's method the tertiary classifier uses only three of the six binary classifiers. In the proposed methods of this paper all the six binary classifiers are applied to form the tertiary classifier so that to give a completeness to the results. Both single window as well as multiple window schemes was tested for getting the best results.

Based on the result of performance comparison of binary classifiers with previous studies, the optimized encoding scheme of this study, the combined matrix of orthogonal and BLOSUM62 matrix, was not satisfactory. However, with the use of new tertiary classifiers of this study, the performance was enhanced. This has been proved by simulation results of 7-fold data sets with Hu’s method.

2. PROTEIN SECONDARY STRUCTURE PREDICTION

2.1. Proteins

Proteins are one of the most basic components in all organisms. They form the basis of cellular and molecular life and affect the structural and functional characteristics of the various cells and genes significantly. On the smallest level, proteins are made up of linear sequences of twenty natural amino acids joined together by peptide bonds. This is known as the primary structure of a protein. This sequence determines the next levels of structures that are formed. Change in a single acid in a critical area of protein can alter the biological function. The secondary structure of a protein is the folding or coiling of the peptide chain. The tertiary structure is the three dimensional shape of the polypeptide chain. The quaternary structure is the final dimensional structure formed by all the polypeptide chains making up a protein.

Knowing amino acid sequences is important for several reasons [8]. First, knowledge of the sequence of a protein is usually essential to elucidating its mechanism of action, such as the catalytic mechanism of an enzyme. Moreover, proteins with novel properties can be generated by varying the sequence of known proteins. Second, amino acid sequence determine the three dimensional structures of proteins. Amino acid sequence is the link between the genetic message in DNA and the three-dimensional structure that performs a protein's biological function. Analyses of relations between amino acid sequences and 3-D structures of proteins are uncovering the rules that govern the folding of polypeptide chains.

Third, sequence determination is a component of molecular pathology, a rapidly growing area of medicine. Alterations in amino acid sequence can produce abnormal function and disease. Severe and sometimes fatal diseases, such as sickle-cell anemia and cystic fibrosis, can result from a change in a single amino acid within protein. Fourth, the sequence of a protein reveals much about its evolutionary history. Proteins resemble one another in amino acid sequence only if they have a common ancestor. Consequently molecular events in evolution can be traced from amino acid sequences.

2.2. Protein Structure

Proteins are an important class of biological macromolecules present in all biological organisms, made up of such elements as carbon, hydrogen, nitrogen, oxygen, and sulfur. All proteins are polymers of amino acids. The polymers, also known as peptides consist of a sequence of 20 different L-α-amino acids, also referred to as residues. For chains under 40 residues the term peptide is frequently used instead of protein. To be able to perform their biological function, proteins fold into one, or more, specific spatial conformations driven by a number of non-covalent interactions. In order to understand the functions of proteins at a molecular level, it is often necessary to determine the three dimensional structure of proteins.

Protein has a structural hierarchy containing ‘primary’, ’secondary’, ’tertiary’ and ‘quaternary’. The primary structure defines the linear sequence of assembled amino acids. At each end of the protein, there remains a free amino or carboxyl group, these are referred as N and C terminus which represents the unbounded N and C atom in the amino or carboxyl group respectively.

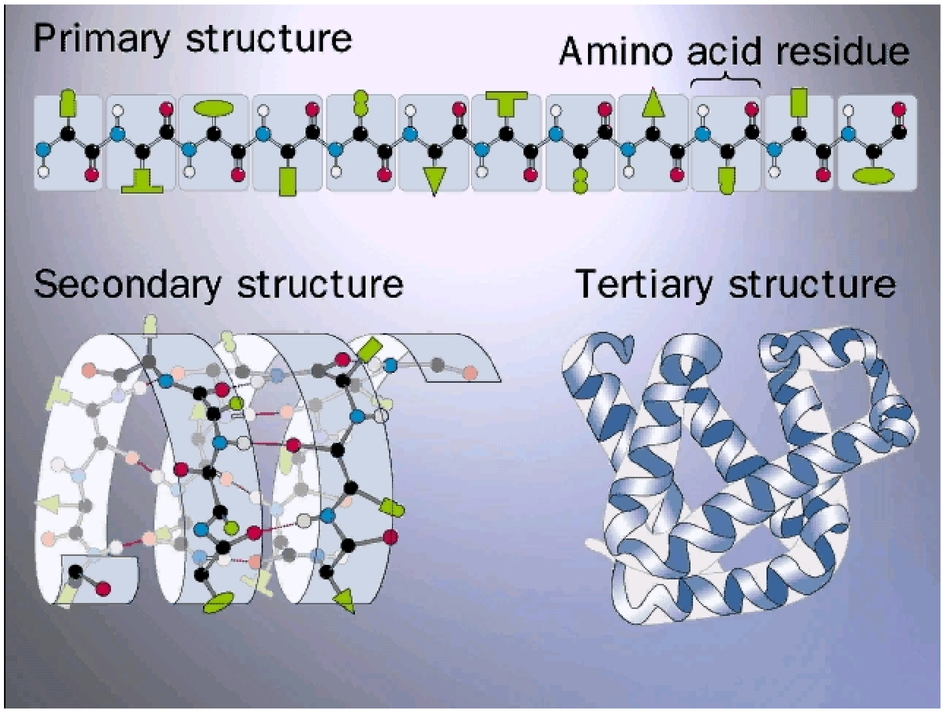


Figure 2-1 Primary and Secondary structure of protein [31].

The tertiary structure is the compact globular structures called domains which are determined and stabilized by chemical bonds and forces, including weak bonds, such as hydrogen bonds, ionic bonds, van der vaals bonds and hydrophobic interactions. This domain is the fundamental unit of tertiary structure, since it forms an independent stable structure and is the basic unit of protein function.

The quaternary structure is the final level of structural hierarchy. Although some proteins are monomeric which consist of only one polypeptide chain, others are multimeric, constructed from several chains. There subunits may work cooperatively, and the functional state of one subunit can depend on the state of the other units. In Figure 2.3, the examples of tertiary and quaternary structures are shown.

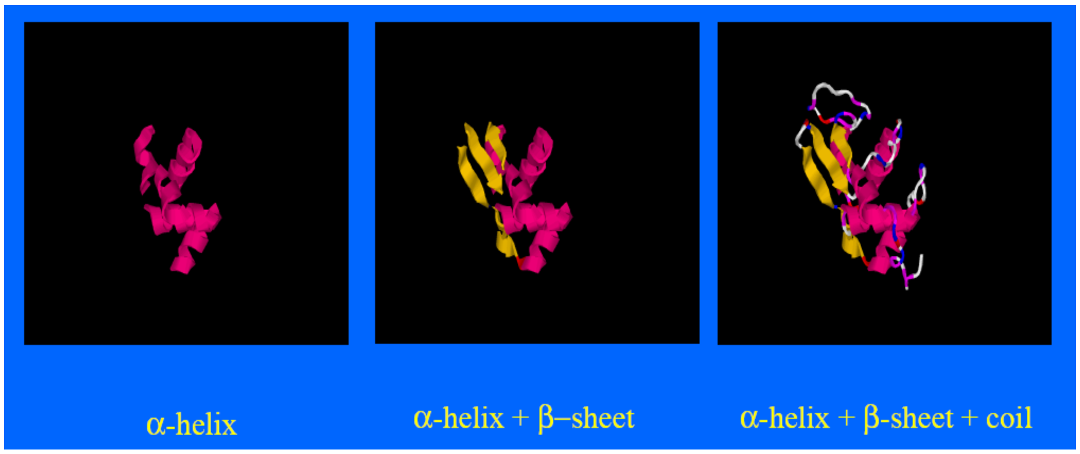


Figure 2-2 Three typical secondary structure elements [31].

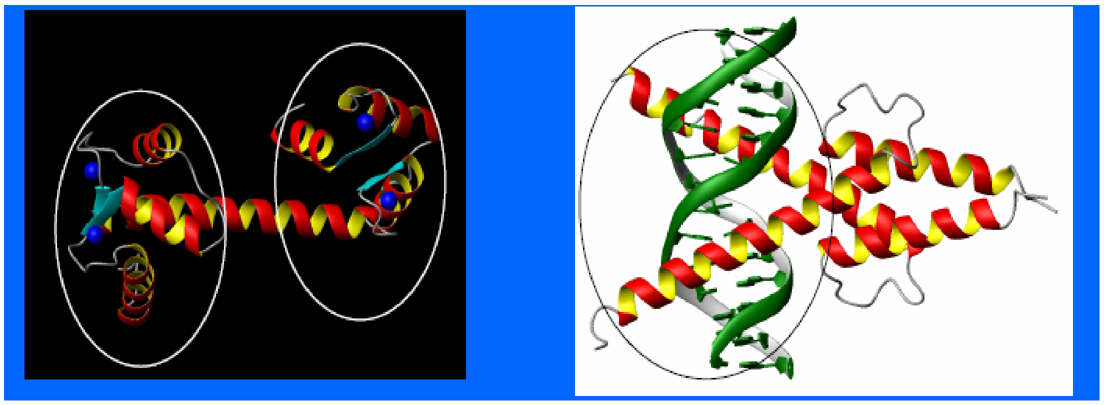


Figure 2-3 Tertiary and Quaternary structure of protein [31].

2.3. Protein Structure Prediction

To be able to find the tertiary structure of the protein, reliable prediction of the secondary structure of a protein is important as the latter leads to the prediction of the former. Protein structure can also be used to infer bio-chemical and biological functional information, and in identification of amino acids that are involved in active site.

The functional properties of proteins depend upon their three dimensional structures. To understand the biological functions of proteins, the structure of a protein from the amino acid sequence should be known beforehand. Therefore, knowledge of the tertiary structure of the protein is a prerequisite for the proper engineering of its function.

A predicted model of a protein can also serve as an excellent basis for identifying amino acid residues involved in the active site and the model can be used for protein engineering, drug design or immunological studies. With this approach, drugs can be developed to cure specific diseases such as sickle cell disease; Parkinson's Alzheimer's disease and may other inherited metabolic disorders.

The current experimental methods for determining the secondary or tertiary structure of proteins are X-ray crystallography, nuclear magnetic resonance (NMR) and electron microscope [8]. However, each method has its limitations because in addition to not being able to characterize most of the transient or stable complexes that exist in a cell, they are also expensive, laborious and time consuming, taking months or even years to complete

2.4. Protein Secondary Structure Prediction

The prediction of secondary structure had been used as a stepping stone to find out the full 3-D structure since it is considerable less complicated than full tertiary structure prediction. The secondary structure of the protein refers to the interactions that occur between CO and NH groups on its amino acids to form folds and loops. Eight elements of the secondary assignment were developed in 1983 [9] and are widely used today, but secondary structure prediction is often limited to three of those: α – helix, extended β- sheet and coil.

Besides the application of the three dimensional structure predictions, secondary structure predictions have other advantageous applications. For example, predicted secondary structure can be used to infer bio-chemical and biological functional information, and in identification of regions of the protein that will likely to undergo structural changes. Figure 2.4 shows the possible applications of protein secondary structure prediction.

There are four main approaches of secondary structure prediction and those are

- Empirical Statistical Methods
- Nearest Neighbor Method
- Hidden Markov Model Methods
- Machine learning Methods

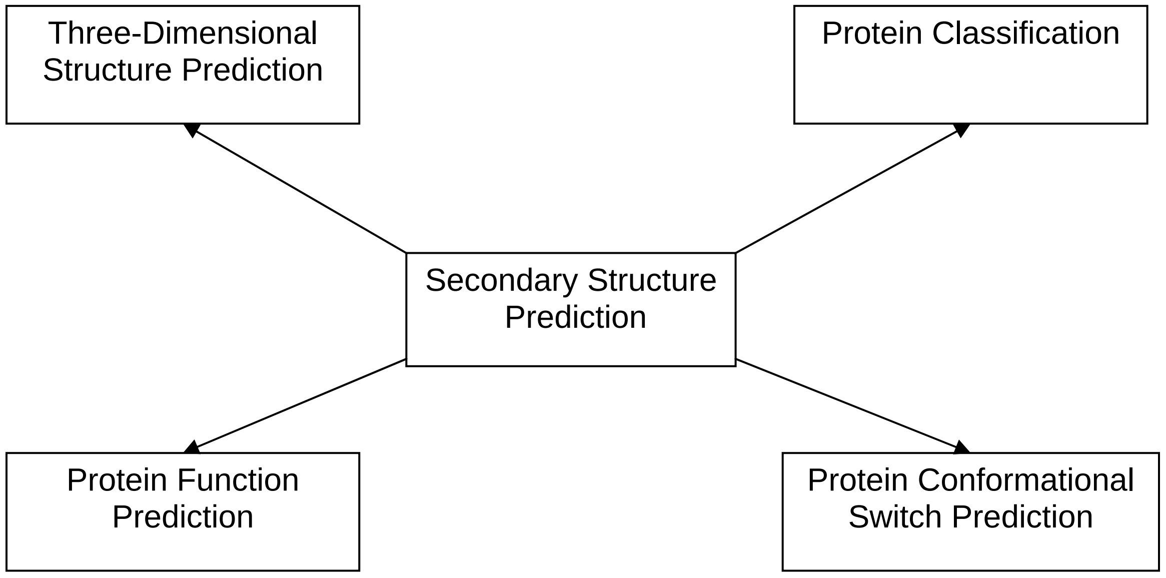

Three-Dimensional Structure Prediction
Protein Classification
Secondary Structure Prediction
Protein Function Prediction
Protein Conformational Switch Prediction

Figure 2-4 Application of Protein Secondary Structure Prediction.

2.4.1. Empirical Statistical Methods

The empirical statistical methods of the protein structure prediction are among the first programs to predict protein structure from amino acid sequence. The example of these empirical methods are Chou-Fasman method and GOR method of secondary structure prediction that use values obtained from known protein structures, which at the time of development of each method were quite small.

The Chou-Fasman method, developed in 1974, was the first algorithm designed for prediction the secondary structure of globular proteins [10]. This method assigns each individual amino acid a value based on their frequency of being in an : α – helix or β- strand that categorizes them into six groups based on frequency for each of the two secondary structure elements. A query sequence is the scanned of region where three of five amino acids gave a high probability of being in a β- strand, or four of six amino acids have a high probability of being in an α – helix, and the probability of amino acids in either direction are calculated for being in that type of structure until the prediction value drops below a specific value. After assigning α – helix and β- sheet to the appropriate regions of the sequence according to the rules for this method, remaining regions are analyzed for coils using a more complex set of rules.

A second statistical method is that of Garnier, Osguthorpe and Robson (GOR) [11]. This method calculates probability values for a specific amino acid based in the adjacent amino acids up to eight residues away using principle of information theory. The GOR method was developed in 1978 and has been updated many times since then, the most recent version based on the current database of 513 domains recommended by Cuff and Barton [12].

The accuracy of these statistical methods was greatly improved in 1993 when Rost and Sander included multiple sequence profiles in theory secondary structure prediction programs [1]. These algorithms assume that higher similar evolutionary related sequence shares at least some secondary structure elements, particularly at conserved sites.

Recently, the program PSI-BLAST [5], which can identify more distant relationships, has been used to widen the profile by including more evolutionarily related matches and increase accuracy [13].

2.4.2. Nearest Neighbor Methods

Nearest neighbor based methods differ from other approaches such that they predict the secondary structure of a target protein using local sequence similarity to segments of known proteins, usually through a sliding window, even when overall target proteins sequence differ substantially from the reference proteins.

This approach benefits from availability of numerous similarity matches or from several highly identical matches of known structures. Currently the two most popular nearest neighbor prediction servers are NNSSP [14] and Preator [15].

The NNSSP server, which includes evolutionary information through multiple sequence alignments, has 72.2% correct predictions. The Predator [15] adopts local pair-wise-alignment of the target sequence as opposed to the multiple sequence alignment. The carefully selected alignment is derived from known structures.

2.4.3. Models Methods Based on Hidden Markov

Hidden Markov Models have also been used to predict secondary structure using the same concept as the nearest neighbor technique. Once a multiple sequence alignment profile is built using short segments of similar sequences with known structure hidden Markov Models are generated in a structure context that is then used to predict the structure of the unknown protein [16]. The program HMMSTR developed by Bystroff et al. uses this method claiming 74.3% accuracy [17].

2.4.4. Neural Network Methods

This method is based on the operation of synaptic connections in neurons of the brain, where input is processed in several levels and mapped to a final output. The neural network is often "trained" to map specific input signals to a desired output. In the secondary structure prediction of neural networks, input is a sliding window with 13-17 residue sequence. Information from the central amino acid of each input window is modifies by a weighting factor, summed and sent to a second level, termed the hidden layer, where the signal is transformed into a number close to either 1 or zero, and then sent to three output units representing each of the possible secondary structures [16]. The output units each weigh the signal again and sum them, and transform them into either a 1 or a 0. An output signal close to 1 for a secondary structure unit indicates that the secondary structure of that unit is predicted and an output signal close to 0 indicated that it is not predicted.

Neural network are trained by adjusting the values of the weights that modify that signals using a training set of sequences with known structure. The neural network algorithm the weight values until the program has been optimized to correctly predict the most residues in the training set.

Typical neural network algorithms include PHD program developed by Rost and Sander [1] which is trained in a test set of profiles of multiple sequences and claims 72.2% accuracy. Recently, this accuracy was shown to be increased to 75% when larger databases and PSI-BLAST [5] was used to create the training set [14].

2.4.5. Summary of Prediction Methods

In Table 2.1, the performance of various prediction techniques is compared. In some cases, the verified performance turned out to be lower than the performances reported by the authors. One reason for the difference in the results could be the different sets used for training and testing networks.

Table 2-1 Performance comparison of various prediction methods [32].

Type	Method	Year	Generation*	Q3%Claimed	Q3%Verified
Statistical	Chou and Fasman	1974	First	77	50
	Garnier et al. (Heiler	1978	First	57	56
Neural Network	Qian and Sejnowski	1988	Second	64.3	
	Rost and Sander (PHD)	1994	Third	72.2	71.6
	Pollastri et al. (Sspro)	2002	Third	76.0	74.5
Nearest Neighbor	Sal.&Solovyev(NNSSP)	1995	Third	72.2	
	Frish.& Argos (Predator)	1997	Third	75.0	
Hidden Markov Models	Karplus et al. (SAM-T99)	1998	Third	-	74.9
	Bystroff et al. (Bystroff)	2000	Third	74.3	

Note:

*Generation (Rost 2000):

First: only single residue statistics used

Second: sliding windows and large database applied

Third: long range interaction through evolutionary information introduced.

3. SUPPORT VECTOR MACHINES

3.1. Overview of SVM

Support Vector Machines (SVM) are learning systems that use a hypothesis space of linear function in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory. This learning strategy introduced by Vapnik and co-workers is a principled and very powerful method that in the few years since its introduction has already outperformed most other systems in a wide variety of applications.

In supervised learning the learning machine is given a training set of examples (or inputs) with associated labels (or output values). Usually the examples are in the form of attributes vectors, so that the input space is a subset of R^n. Once the attributes vectors are available, a number of sets of hypotheses could be chosen for the problem. Among these, linear functions are best understood and simplest to apply. Traditional statistics and the classical neural networks literature have developed many methods for discriminating between two classes of instances using linear functions.

The problem of learning from data has been investigated by philosophers throughout history under the name of 'inductive inference'. Although this might seem surprising today, it was not until 20th century that pure induction was recognizes as impossible unless one assumes some prior knowledge. The development of learning algorithm became an important sub field of artificial intelligence, eventually forming the separate subject area of machine learning.

Kernel representations offer an alternative solution by projecting the data into a high dimensional feature space to increase the computational power of the linear learning machines.

Another attraction of kernel methods is that the learning algorithms as theory can largely be decoupled from the specifics of the application area, which must simply be encoded into the design of an appropriate kernel function. Hence the problem of choosing architecture for a neural network application is replaced by the problem of choosing a suitable kernel for a Support Vector Machines. The introduction of kernel greatly increases the expressive power of the learning machines while retraining the underlying linearity that will ensure that learning remains tractable. The increased flexibility however, increases the risk of over fitting as the choice of separating hyperplane becomes increasingly ill-posted due to the number of degrees of freedom. Successfully controlling the increased flexibility of kernel-induced feature spaces requires a sophisticated theory of generalization, which is able to precisely describe which factors have to be controlled in the learning machines in order to guarantee good generalization. There is a remarkable family of bounds governing the relation between the capacity of a learning machines and its performance. The theory grew out of consideration of under what circumstances and how quickly, the mean of some empirical quantities converges uniformly, as the number of data points increases, to the true mean [7].

Since SVM approach has a number of superior such as effective avoidance of over-fitting, the ability to handle large feature spaces, information condensing of the given data set, it has been successfully applied to a wide range of pattern recognition problems, including isolated handwritten digit recognition, objective recognition, speaker identification, and text categorization, etc [3].

3.2. Linear Classification [18]

Binary classifier is frequently implemented by using a real-valued function $f: X \subseteq \Re^n \rightarrow \Re$ in the following way: the input $x = (x_1,....,x_n)'$ is assigned to the positive class, if $f(x) \geq 0$, and otherwise to the negative class. If we consider the case where $f(x)$ is a linear function of $x \in X$, so that it can be written as

$$f(x) = w \bullet x + b$$

$$= \sum_{i=1}^{n} w_i x_i + b$$

Where, $(w, b) \in \Re^n \times \Re$ are the parameters that control the function and the decision rule given by $\operatorname{sgn}(f(x))$. And these parameters must be learned from the data.

If we interpret this hypothesis geometrically, input space X is split into two parts by the hyperplane defined by the equation $w \bullet x + b = 0$. For example, in Figure 3.1, the hyperplane is the dark line, with the positive region above and the negative region below. The vector w defines a direction perpendicular to the hyperplane, while varying the value of b moves the hyperplane parallel to itself. And these quantities are referred as the weight vector and bias which are the terms borrowed from the neural networks literature.

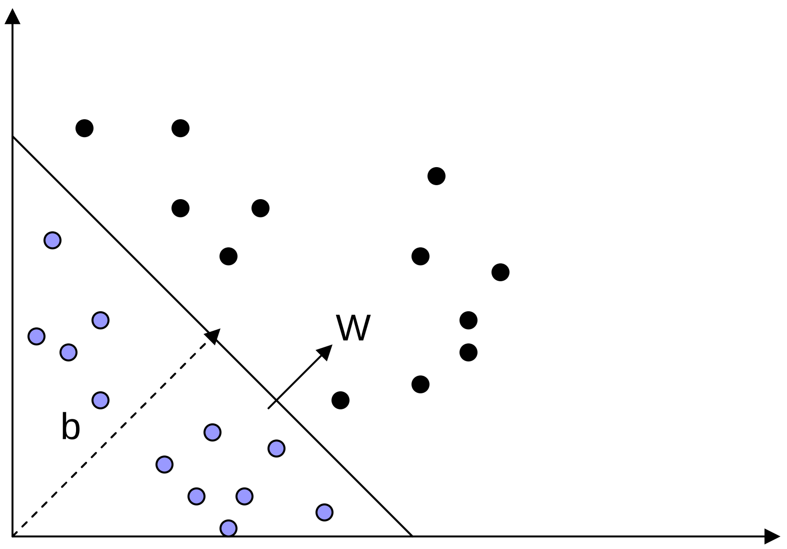


Figure 3-1 A separating hyperplane for a 2-d training set [18].

3.3. Support Vector Algorithm

3.3.1. Linear SVM – Maximum Margin Classifier

The training data is defined as a set { x_i,y_i } , 1 = 1,….1 , y_i *є { – 1, 1 }* , x_i *є* $\boldsymbol{R}^d$. Suppose we have some hyperplane which separates the positive from negative examples (a ‘separating hyperplane ‘). The points x which lie on the hyperplane satisfy w. x + b = 0, where w is normal to the hyperplane, |b| / ||w|| is the perpendicular distance from the hyperplane to the origin, and ||w|| is the Euclidean norm of w. Let d_+ (d_-) be the shortest distance from the separating hyperplane to the closest positive (negative) examples. Define the ‘margin’ of a separating hyperplane to be $d_+ + d_-$. For the linearly separable case, the support vector algorithm simply looks for the separating hyperplane with the largest margin. This can be formulated as follows: suppose that all the training data satisfy the following constrain [18].

$$x_i \bullet w + b \geq +1 \quad for \quad y_i = +1 \tag{3.1}$$

$$x_i \bullet w + b \leq -1 \quad for \quad y_i = -1 \tag{3.2}$$

These can be combined into one set of inequalities:

$$y_i \left(x_i \bullet w + b\right) \; - \; 1 \; \geq 0 \quad for\ all\ i \tag{3.3}$$

Now consider the point for which the equality in Eq. (3.1) holds (requires that there exists such a point in equivalent to choosing a scale for w and b). These points lie on the hyperplane H_1 : x_i . w + b =1 with normal w and perpendicular distance from the origin |1 – b| / ||w||. Similarly, the points for which the equality in Eq (3.2) holds lie on the hyperplane H_2: x_i. w + b = – 1, with normal again w, and the perpendicular distance from the origin | – 1 – b| / ||w||. Hence $d_+ = d_- =$ 1/||w|| and the margin is simple 2/||w||. Note that H_1 and H_{12} are parallel (they have the same normal) and that no training points fall between them. Thus we can find the pair of hyper planes which gives the maximum margin by minimizing $||w||^2$, subject to the constraints (3.3).

Thus we expect the solution for a typical two dimensional case to have the form shown in Figure 2.5. Those training points for which the equality in Eq. (3.3) holds (i.e. those which wind up lying on one of the hyper planes H_1, H_1.), and whose removal would change the solution found, are called support vectors; they are indicated in Figure 3.1 by extra circles.

Now Lagrangian formulation of the problem is done. There are two reasons for doing this. The first is that the constraints (3.3) will be replaced by constraints on the Lagrange multipliers themselves, which will be much easier to handle [19]. The second is that in this reformulation of this problem, the training data will only appear (in the actual property and test algorithms) in the form of dot products between vectors. This is a crucial property which will allow us to generalize the procedure to the nonlinear case.

Thus, we introduce positive Lagrange multiplier $\alpha_i, i = 1,......, 1,$ one for each of the inequality constraints (3). Recall that the rule is that for constraints of the form $c_i \geq 0$, the constraint equations are multipliers and subtracted from the objective function, to form Lagrangian. For equality constraints, the Lagrange multipliers are unconstrained. This gives Lagrangian:

$$L_P \equiv \frac{1}{2}\|w\|^2 - \sum_{i=1}^{l} \alpha_i \, y_i \, (x_i \bullet w + b) \quad + \sum_{i=1}^{l} \alpha_i \tag{3.4}$$

We must now minimize L_P with respect to w, b, and simultaneously require that the derivatives of L_P with respect to all the α_i vanish, all the subject to the constraints $\alpha_i \geq 0$ (let's call this particular set of constraints C_1). Now this is a convex quadratic programming problem, since the objective function is itself convex, and those points which satisfy the constraint also from the convex set (any linear constraint defines a convex set and a set of N simultaneously linear constraint defines the intersection of N convex sets which is also a convex set). This means that we can equivalently solve the following dual problem: maximize L_P, subject to the constraint that the gradient of L_P with respect to w and b vanish, and subject also to the constraint that the $\alpha_i \geq 0$ (let's call that particular set of constraints C_2). This particular dual formulation of the problem is called the Wolfe dual. It has the property that the maximum of L_P, subject to constraint C_2, occurs at the same value of w and b and $\square_i$ as the minimum of L_P, subject to constraints C_1 [19].

Requiring that the gradient pf L_P with respect to w and b vanish give the conditions:

$$w = \sum_i \alpha_i \, y_i \, x_i \tag{3.5}$$

$$\sum_i \alpha_i \, y_i = 0 \tag{3.6}$$

Since these are equality constraints in the dual formulation, we can substitute them into Eq (3.4) to give

$$L_D \quad \equiv \quad \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \, \alpha_j \, y_i \, y_j \, x_i \, x_j \tag{3.7}$$

Note that we have now given the Lagrangian different labels (P for primal, D for dual) to emphasize that the two formulations are different: L_P and L_D arise from the same objective function but with different constraints; and the solution is found by minimizing L_P or by maximizing L_D. Note also that if we formulate the problem with b = 0, which amounts to requiring that all hyper planes contain the origin, the constraint (3.6) does not appear. This is a mild restriction for high dimensional spaces, since it amounts to reducing the number of degree of freedoms by one.

Support vector training (for separable linear case) therefore amounts to maximizing L_D with respect to the α_i, subject to constraint (3.6) and positively of the α_i, with solution given by (3.5). Notice that there is a Lagrange multiplier α_i for every training point. In the solution, those points for which $\alpha_i > 0$ are called "support vectors", and lie on one of the hyper planes H_1, H_2. All other training point have $\alpha_i = 0$ and lie either on H_1 or H_2 (such that equality in Eq. (3.3) holds), or on the side of H_1 or H_2 such that the strict inequality in Eq. (3.3) holds. For these machines support vectors are critical elements of training set. They lie closest to the decision boundary; if all other trainings points are removed (or moved around, but so as not to cross H_1 or H_2.), and the trainings was repeated, the same separating hyperplane would be found.

3.3.2. The Non-Separable Case

The above algorithm for separable data, when applied to non-separable data, will find no feasible solution: this will be evidenced by the objective function) i.e. the dual Langrangian) growing arbitrarily large. So how can we extend these ideas to handle non-separable data? We would like to relax the constraints (3.1) and (3.2), but only when necessary, that is, we would like to introduce a further cost (i.e. an increase in the primal objective function) for doing so. This can be done by introducing positive slack variables $\xi_i, i = 1,...1$ in the constraints, which then become:

$$x_i \bullet w + b \geq +1 - \xi_i \quad for\ y_i = +1 \tag{3.8}$$

$$x_i \bullet w + b \leq -1 + \xi_i \quad for\ y_i = -1 \tag{3.9}$$

Thus, for an error to occur, the corresponding ξ_i must exceed unity, so $\sum_i \xi_i$ is an upper bound on the number of training errors. Hence a natural way to assign an extra cost for error is to change the objective function to be minimized from $||w||^2/2$ to $||w||^2/2 + C(\sum_I \xi_i)^k$, where C is a parameter to be chosen by the user, a larger C corresponding to assigning a higher penalty to error. [19] As it stands, this is a convex programming problem for any positive integer k; for j=2 and k =1, it is also a quadratic programming problem and the choice k = 1 has further advantage that neither the ξ_i, nor their Lagrange multiplier, appears in the Wolfe dual problem, which becomes:

Maximize:

$$L_D \equiv \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j x_i x_j \tag{3.10}$$

Subject to:

$$0 \le \alpha_i \le C_i \tag{3.11}$$

$$\sum_i \alpha_i y_i = 0 \tag{3.12}$$

The solution is again given by

$$w = \sum_i \alpha_i y_i \sum_{i=1}^{N_S} \alpha_i y_i x_i \tag{3.13}$$

Where N_s is the number of support vectors. Thus the only difference from the optimal hyperplane case is that the α_i now have an upper bound of C. The situation is summarized schematically in Figure 3.2.

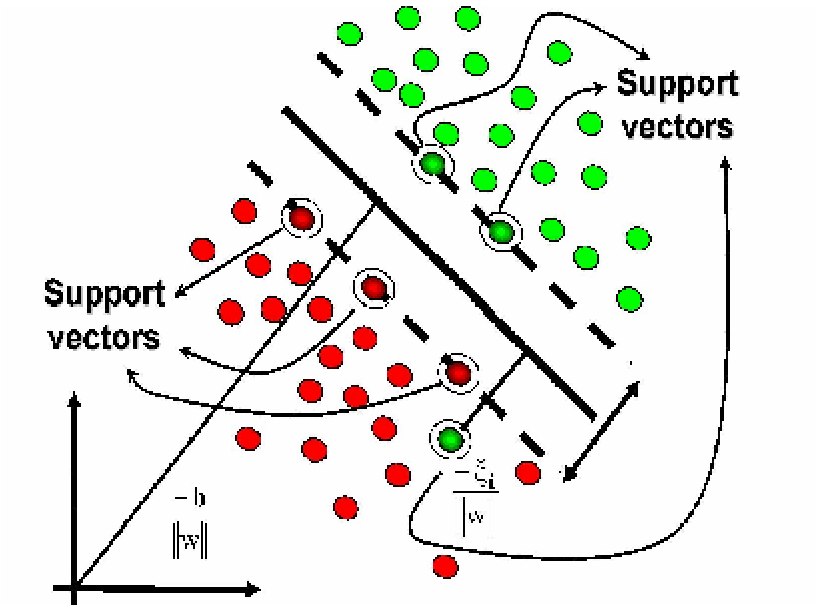


Figure 3-2 Linear separating hyperplane for non-separable case.

We will need the Karush-Kuhn-Tucker condition for the primal problem. The primal Lagrangian is

$$L_P = \frac{1}{2}\|w\|^2 + C\sum_i \xi_i - \sum_i \alpha_i \{y_i(x_i \bullet w + b) - 1 + \xi_i\} - \sum_i \mu_i \xi_i \qquad (3.14)$$

Where the μ_i are the Lagrange multipliers introduced to enforce positivity of the ξ_i. The KKT condition for the primal problem are therefore (note I runs from 1 to the number of training points, and v from 1 to dimension of the data)

$$\frac{\partial L_P}{\partial w_v} = w_v - \sum_i \alpha_i y_i x_{iv} = 0 \qquad (3.15)$$

$$\frac{\partial L_P}{\partial b} = -\sum_i \alpha_i y_i = 0 \qquad (3.16)$$

$$\frac{\partial L_P}{\partial \xi_i} = C - \alpha_i - \mu_i = 0 \tag{3.17}$$

$$y_i \left(x_i \bullet w + b\right) - 1 + \xi_i \geq 0 \tag{3.18}$$

$$\xi_i \geq 0 \tag{3.19}$$

$$\alpha_i \geq 0 \tag{3.20}$$

$$\mu_i \geq 0 \tag{3.21}$$

$$\alpha_i \left\{ y_i \left(x_i \bullet w + b\right) - 1 + \xi_i \right\} = 0 \tag{3.22}$$

$$\mu_i \, \xi_i = 0 \tag{3.23}$$

As before, we can use KKT complementarily conditions Eqs (3.22) and (3.23), to determine the threshold b. Note that Eq. (3.17) shows that $\xi_i = 0$ if $\alpha_i < C$. Thus we can simply take any point for which $0 < \alpha_i < C$ to use (3.22) (with $\xi_i = 0$) to compute b [18].

3.3.3. Nonlinear SVM - Kernel Method

The soft margin classifier is an extension of linear SVM. The kernel method is a scheme to find the nonlinear boundaries. The concept of the kernel method is transformation of the vector space to a higher dimensional space. As can be seen from Figure 2.7, by transforming the vector space from two-dimensional to three-dimensional space, the non-separable vectors can be separated.

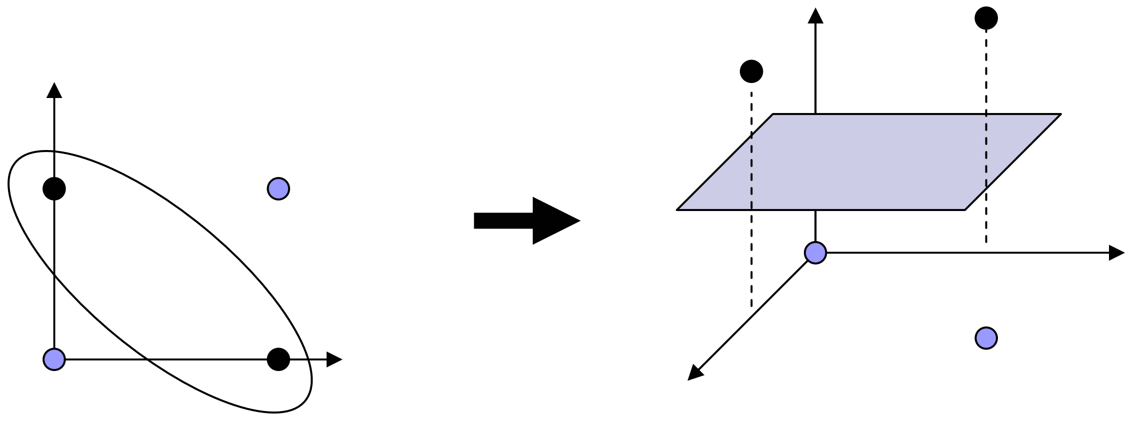


Figure 3-3 Transformation to higher dimensional space

3.4. SVM Software

The prediction of protein secondary structure is done using SVM^{light} software. SVM^{light} software is the implementation of Vapnik's Support Vector Machine (Vapnik 1995) for the problem of pattern recognition, regression and ranking function. SVM^{light} software consists of two parts, the first part i.e. is the svm_learn part takes care of the learning module and the second part svm_classify part does the classification of the data after training.

The input data to both the parts should be given in the following format

<line> .=. <target> <feature>:<value> <feature>:<value> …

<target> .=. +1 | -1 | 0 | <float>

<feature> .=. <integer> | "qid"

<value> .=. <float>

The target value and each of the feature/value pairs are separated by space character. Feature/value pairs must be ordered by increasing feature number. Features with value zero can be skipped. For classification, the target value denotes the class of the example +1 as the target value marks a positive example, -1 negative example respectively. So, for example, the line

```
-1 1:0.43 3:0.12 2345:0.9
```

Specifies a negative example for which feature number 1 has value o.43 feature number 3 has value 0.12 feature number 2345 has the value 0.9 and all the other features have value 0. the order of the predictions is the same as in the training data [20].

4. RELATED SVM-BASED METHODS

4.1. Secondary Structure Assignment

The secondary structure is converted from the experimentally determined tertiary structure by DSSP [9], STRIDE [15] or DEFINE . In this study, the DSSP scheme is used since it is the most generally used secondary structure prediction method.

The DSSP classifies residues into eight different secondary structure classes: H (α-helix), G (3_{10} – helix), I (π-helix), E (β-strand), B (isolated β-bridge), T (turn), S (bend), and – (rest). In this study, these eight classes are reduced into three regular classes based on the following Table 3.1

Table 4-1 8-to-3 state reduction method in secondary structure assignment

DSSP Class	8-state symbol	3-state symbol	Class name
3_{10} – helix α-helix π-helix	G H I	H	Helix
β-strand	E	E	Sheet
isolated β-bridge Bend Turn Rest (connection region)	B S T -	C	Loop

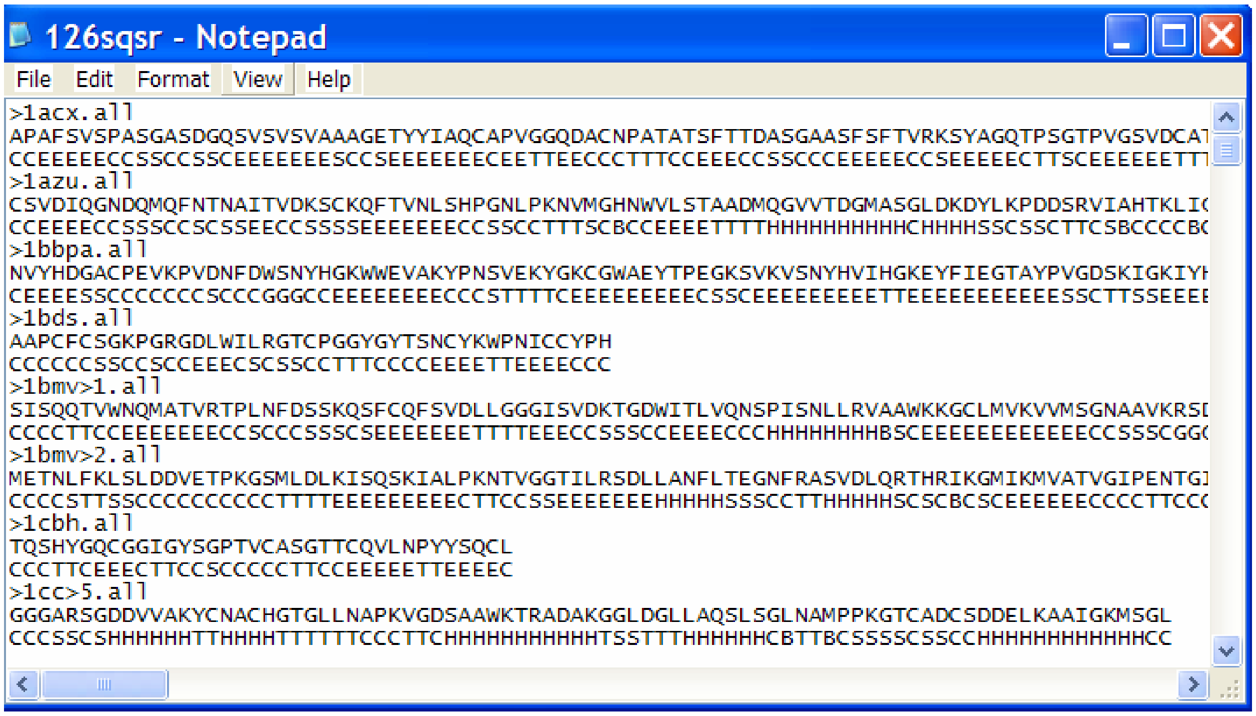
4.2. Training and Testing Data Sets

For comparing the results of this study with previously published results [6], RS 126 data set is used.

The RS 126 data set is proposed by Rost & Sander [1] and according to their definition, it is non-homologous set. They used percentage identity to measure the homology and defines non-homologous as no two proteins in the set share more than 25% sequence identity over a length of more than 80 residues.

For each data set, the seven fold cross validation is done [1] [3] [6]. In the seven-fold cross validation test, one subset is chosen for testing and remaining 6 subsets are sued for training and this process is repeated until all the subsets are chosen for the testing.

In Figure 4.1, RS 126 set is displayed



```
>1acx.all
APAFSVSPASGASDGQSVSVSVAAAGETYYIAQCAPVGGQDACNPATATSFTTDASGAASFSFTVRKSYAGQTPSGTPVGSVDCAT
CCEEEEEECCSSCCSSCEEEEEEEESCCSEEEEEEEECEETTEECCCTTTCCEEECCSSCCCEEEEEECCSEEEEEECTTSCEEEEEEETTT
>1azu.all
CSVDIQGNDQMQFNTNAITVDKSCKQFTVNLSHPGNLPKNVMGHNWVLSTAADMQGVVTDGMASGLDKDYLKPDDSRVIAHTKLI
CCEEEEECCSSSCCSCSSEECCSSSSEEEEEEEECCSSCCTTTSCBCCEEEETTTTHHHHHHHHHHCHHHHSSCSSCTTCSBCCCCB
>1bbpa.all
NVYHDGACPEVKPVDNFDWSNYHGKWWEVAKYPNSVEKYGKCGWAEYTPEGKSVKVSNYHVIHGKEYFIEGTAYPVGDSKIGKIY
CEEEESSCCCCCCCSCCCGGGCCEEEEEEEEECCCSTTTTCEEEEEEEEECSSCEEEEEEEEEETTEEEEEEEEEEEESSCTTSSEEEE
>1bds.all
AAPCFCSGKPGRGDLWILRGTCPGGYGYTSNCYKWPNICCYPH
CCCCCCSSCCSCCEEECSCSSCCTTTCCCCEEEETTEEEECCC
>1bmv>1.all
SISQQTVWNQMATVRTPLNFDSSKQSFCQFSVDLLGGGISVDKTGDWITLVQNSPISNLLRVAAWKKGCLMVKVVMSGNAAVKRSD
CCCCTTCCEEEEEEEECCSCCCSSSCSEEEEEEEETTTTEEECCSSSSCCEEEEECCCHHHHHHHHBSCEEEEEEEEEEEEEECCSSSCGG
>1bmv>2.all
METNLFKLSLDDVETPKGSMLDLKISQSKIALPKNTVGGTILRSDLLANFLTEGNFRASVDLQRTHRIKGMIKMVATVGIPENTG
CCCCSTTSSCCCCCCCCCCTTTTEEEEEEEEEECTTCCSSEEEEEEEEHHHHHSSSCCTTHHHHHSCSCBCSCEEEEEEECCCCTTCCC
>1cbh.all
TQSHYGQCGGIGYSGPTVCASGTTCQVLNPYYSQCL
CCCTTCEEECTTCCSCCCCCTTCCEEEEETTEEEEC
>1cc>5.all
GGGARSGDDVVAKYCNACHGTGLLNAPKVGDSAAWKTRADAKGGLDGLLAQSLSGLNAMPPKGTCADCSDDELKAAIGKMSGL
CCCSSCSHHHHHHTTHHHHTTTTTTCCCTTCHHHHHHHHHHHHTSSTTTHHHHHHCBTTBCSSSSCSSCCHHHHHHHHHHHHCC
```

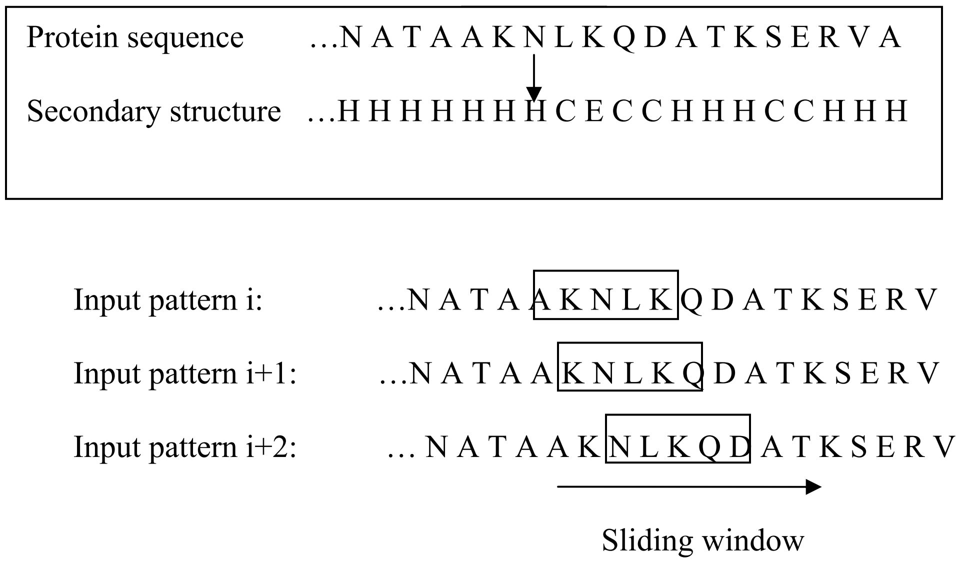
Figure 4-1 RS 126 data sets

4.3. Data Pre-Processing

4.3.1. Sliding Window Scheme

To train the SVM with protein sequence and structural information, a sliding window scheme is sued [6]. In this sliding scheme, a window becomes one training pattern for predicting the structure of the residue at the center of the window. And in this training pattern, the information about the local interactions among neighboring residues is embedded.

Figure 4.2 shows an example of this scheme with window size of 5. Here to predict the structure of amino acid 'N', the sequence 'AKNLK' goes together as one input pattern. To predict structure of 'L', the next amino acid, the window slides down to the next group of sequence, 'KNLKQ' and so on.



Protein sequence …N A T A A K N L K Q D A T K S E R V A
Secondary structure …H H H H H H H C E C C H H H C C H H H
Input pattern i: …N A T A A K N L K Q D A T K S E R V
Input pattern i+1: …N A T A A K N L K Q D A T K S E R V
Input pattern i+2: … N A T A A K N L K Q D A T K S E R V
Sliding window

Figure 4-2 Sliding window scheme with window length of 5

4.3.2. Orthogonal Input Profile

The feature value of each amino acid residue in a window means the weight (costs) of each residue in a pattern. In this study all weight assignment schemes are not tested. The test results from Hu's method are used to select the best scheme. In Hu's study orthogonal method is used as reference for comparison with different encoding scheme.

Among the different schemes explained in the previous studies, the first simplest way is to use the orthogonal encoding which assigns a unique binary vector to each residue, such as (1, 0, 0, …), (0, 1, 0, …), (0, 0, 1 …) so on.[6] In this method, the weights of all the residues in a window are assigned to 1 equally. The method is explained as follows here for simplicity only single window encoding scheme is explained. For multiple windows the same technique is followed for all the elements with in the window and target is the center element of the middle window.

Figure 4.3 shows the sample training data to which orthogonal encoding method is applied as an input profile with window size 5. In Figure 3.5, the value of the first column, {-1, +1}, are target values of each binary classifier. For example, if the binary classifier is the one which classifies helix or not, and if the structure of the residue from the training data is helix, (i.e. the center value has H, G or I corresponding to its position) the target value becomes +1.

After the target value, indices of each amino acid come out with the weight. At the first row of Figure 3.4, 16,37,41,61 and 84 are indices of each amino acid when the window size equals 5 and the value of 1s next to these are the weights of each amino acid. And these weights are all equal in orthogonal encoding scheme.

Since 20 amino acids take one binary bit each, as shown in Figure 3.5, the dimension of one input pattern becomes as follows:

$$\text{One vector dimension} = (20 \text{ binary bits}) \text{ x } \underbrace{(5 \text{ residues})}_{\text{Window size}}$$

$$= 100$$

Therefore, those indices of 16,37,41,61 and 84 mean,

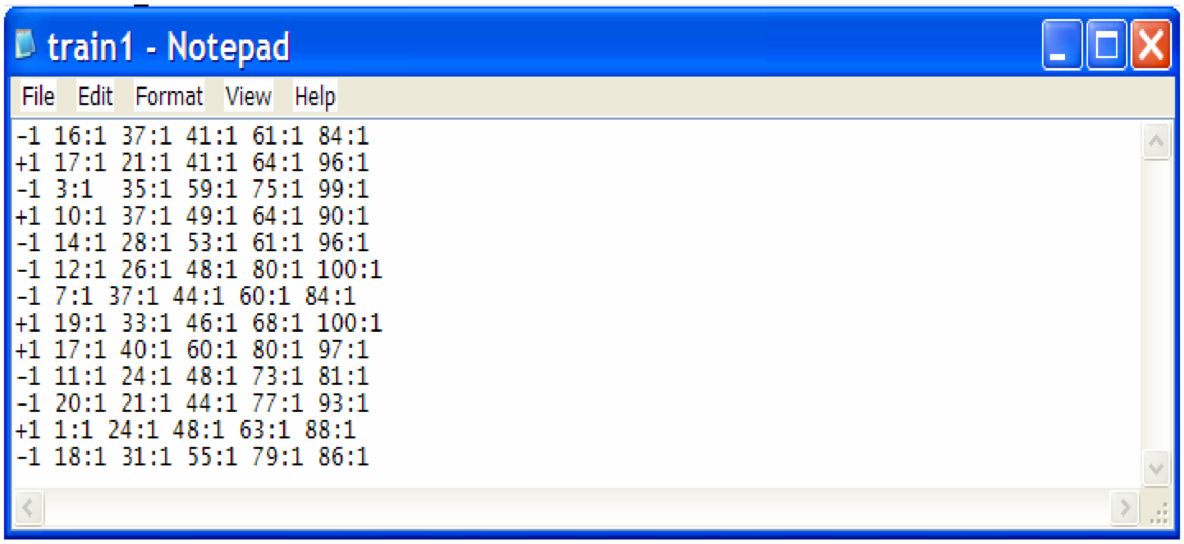
(0, 0, … 1, 0, 0, 0, 0) $\longrightarrow$ S

(0, 0, … 1, 0, 0, 0) $\longrightarrow$ T

(1, 0, 0 …) $\longrightarrow$ A

(1, 0, 0 …) $\longrightarrow$ A

(0,0,0,1,0, ….) $\longrightarrow$ D



```
-1 16:1 37:1 41:1 61:1 84:1
+1 17:1 21:1 41:1 64:1 96:1
-1 3:1  35:1 59:1 75:1 99:1
+1 10:1 37:1 49:1 64:1 90:1
-1 14:1 28:1 53:1 61:1 96:1
-1 12:1 26:1 48:1 80:1 100:1
-1 7:1 37:1 44:1 60:1 84:1
+1 19:1 33:1 46:1 68:1 100:1
+1 17:1 40:1 60:1 80:1 97:1
-1 11:1 24:1 48:1 73:1 81:1
-1 20:1 21:1 44:1 77:1 93:1
+1 1:1 24:1 48:1 63:1 88:1
-1 18:1 31:1 55:1 79:1 86:1
```

Figure 4-3 Sample Training Data with Orthogonal Encoding of Window Size 5.

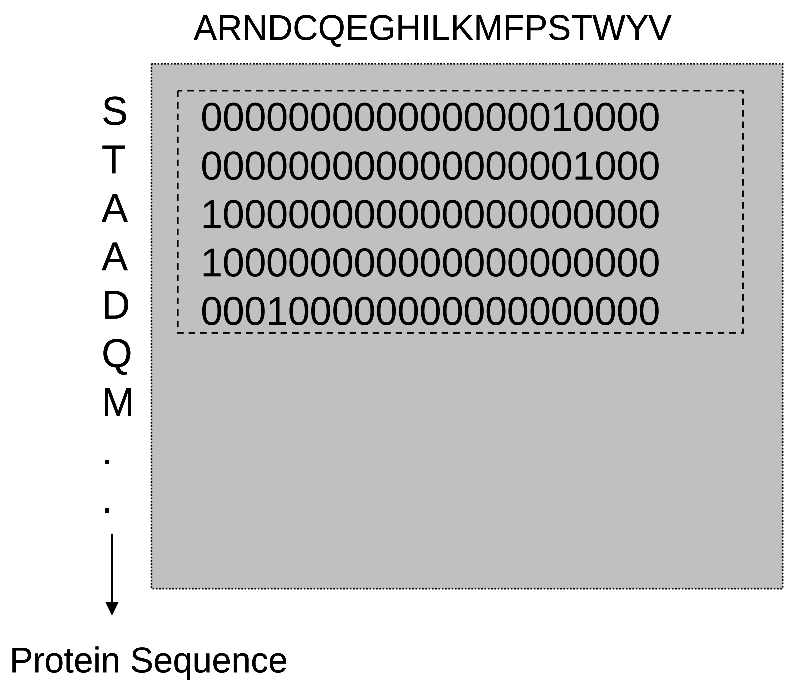


Figure 4-4 An Example of Orthogonal Vector Profile.

Other profiling techniques like ‘Physico-Chemical Property Based Input Profile1’ , ‘Physico- Chemical Property Based Input Profile 2’,’Hydrophobicity Matrix Input Profile’ and ‘BLOSUM62 Matrix Input Profile’ were tested in Hu’s study [6]. Among them only one using BLOSUM62 matrix was shown to have better accuracy compared to orthogonal encoding scheme. So in this study only BLOSUM62 matrix Input Profile is used. Combination of two profiles in hybrid encoding scheme, also discussed in Hu’s study, among them the combination of orthogonal encoding scheme with BLOSUM62 matrix was shown to have the best results. This paper deals mainly with the above mentioned encoding scheme.

4.3.3. BLOSUM62 Matrix Input Profile

The BLOSUM matrices originate from the paper by Henikoff and Henikoff (1992). [21] Their idea was finding a good measure of difference between two proteins specifically for more distantly related proteins.

The value in the BLOSUM62 matrix are ‘log-odds’ scores for the likelihood that a given amino acid pair will interchange. Amino acids with similar physical properties are more likely to replace one another than dissimilar amino acids. So the conservative hydrophobic exchange such as I (Ile) to L (Leu) has a positive score whereas changing I to the hydrophilic residue N (Asn) receives a negative score, which means that this kind of interchange is not likely to occur. [6] More detailed information about the BLOSUM62 matrix is given in Appendix.

In this research, instead of using the orthogonal vector, the BLOSUM62 matrix is applied as an input profile. But since the range of BLOSUM62 matrix value is {-4,11], for the proper encoding of the SVM, the range should be converted to [0, 1] in the preprocessing step. Hence, for this conversion, the following two functions are applied and compared.

Function (4.1) is a simple linear function with range conversion from [-4, 11] to [0, 1] and Function (4.2) is an exponential function. As the BLOSUM62 matrix is a log odd matrix with base 2, the exponential function, such as (4.2), is designed. And if function (4.2) is applied, the values of BLOSUM62 matrix over 6 are assigned to 1 regardless of the final value.

$$f(x) = \frac{1}{15}x + \frac{4}{15} \tag{4.1}$$

$$f(x) = \frac{2^{\frac{x}{2}}}{10} \tag{4.2}$$

Where, x is the value from the raw profile matrix.

4.3.4. Combined Input Profile

To obtain the optimal input profile which offer the most informative feature to predict the secondary structure with high accuracy, the previous input profile are combined together. When more than one encoding scheme is used the weight is applied based on a position inside a window. In other words, even though each amino acid has 20 different 'log odds' scores, those values are always same regardless of the position inside the sliding window. Therefore by assigning different weights based on their position inside the window, the machine could be trained with more specific information.

4.4. Parameter Optimization of the Binary Classifier

To achieve high testing accuracy, a suitable kernel function, its parameter and the regularization parameter C should be properly selected. In Figure 4.5, typical kernel functions are shown. Draper [14] compared the kernel for protein secondary structure prediction with SVM and concluded that the polynomial and Gaussian kernels, also known as RBF kernel, were the best.

Hua and Sun [3] has proved that the Gaussian kernel can provide superior performance in the generalization ability and convergence speed. Therefore, in this study, according to the previous result, Gaussian radial basis function kernel was adopted.

Once the kernel function is selected, the parameter of the kernel function, γ, and the regularization parameter, C which controls the trade-off between complexity and misclassified training example, should be specified by the user.

The soft-margin SVM maximizes margin with minimizing training error simultaneously by solving the following optimization problem

$$Minimize_{\xi,w,b} \qquad w \bullet w \ + \ C\sum_{i=1}^{l}\xi_i$$

$$\text{Subject to } y_i\left(w \bullet x_i \ + \ b\right) \ \geq \ 1-\xi_i, \qquad i=1,...,l$$

$$\xi_i \ \geq \ 0, \quad i=1,...,l$$

Where, x_i represents an input vector, $y_i = +1, or -1$ according to whether x_i is positive class or negative class, l is the number of the training data, and C is the regularization parameter that controls the trade-off between margin and classification error represented by slack variable$\left(\xi_i\right)$.

The corresponding dual quadratic programming problem with the application of a kernel function $K\left(x_i, x_j\right)$ was written as:

$$\textit{Maximize} \quad \sum_i \lambda_i - \frac{1}{2}\sum_{i,j} \lambda_i \; \lambda_j \; y_i \; y_j \; K(x_i, x_j)$$

$$\text{Subject to} \quad 0 \le \lambda_i \le C$$

$$\sum_i \lambda_i \; y_i \quad = \quad 0$$

The dual formulation of the soft margin SVM with regularization parameter C shows that the influence of a single training example is limited by C. And it is known that a large C value is imposes a high penalty to classification errors

For the proper choice of C value and γ value Gaussian radial basis function, $e^{-\gamma\|x-y\|^2}$, the previous studies tested different γ_s and upper bound value of C over their own data sets and selected the pairs which shows the best accuracy [22] [4] [3].

Simple dot product:

$$K(x,y) \;=\; x \bullet y$$

Vovk's polynomial:

$$K(x,y) \;=\; (x \bullet y + 1)^p$$

Radial basis function (RBF):

$$K(x,y) \;=\; e^{-\gamma\|x-y\|^2}$$

Two layer neural network:

$$K(x,y) \;=\; \tanh(kx \bullet y - \partial)$$

Figure 4-5 Typical kernel Functions in SVM

4.5. Binary Classifier Construction

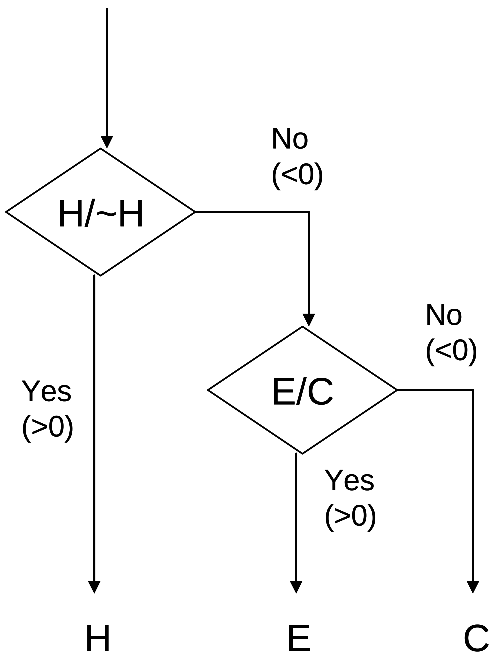
Six SVM binary classifier including three one-versus-rest classifier (‘one’: positive class, ‘rest’: negative class) names H/~H, E/~E and C/~C and three one-versus-one classifier named H/E, E/C, C/H were constructed. For example, the classifier H/E is constructed on the training samples having helices and sheets and it classifies the testing sample as helix or sheet. The programs for constructing the SVM binary classifier were written in the C language.

4.6. Tertiary Classifier Design

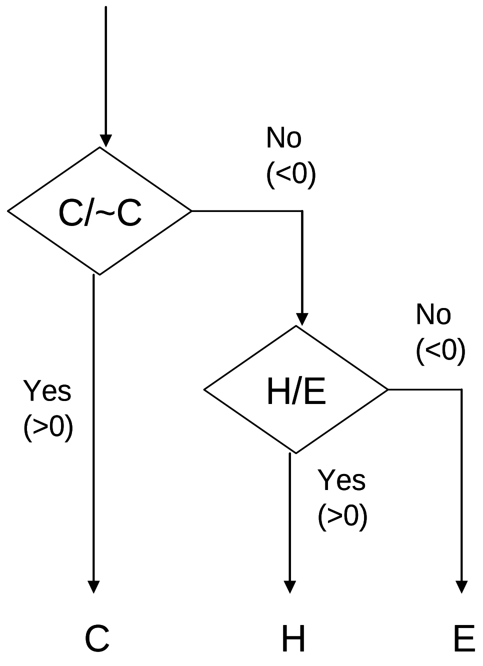
There are many ways to combine the output from the binary classifier for secondary structure prediction. In this research, several tertiary classifiers proposed by previous studies [3] [22] were tested and compared with the new tertiary classifier of this study. Here, the new tertiary classifier is designed based on the results of three one-versus-one binary classifier.

4.6.1. Tree-based tertiary classifier [3]

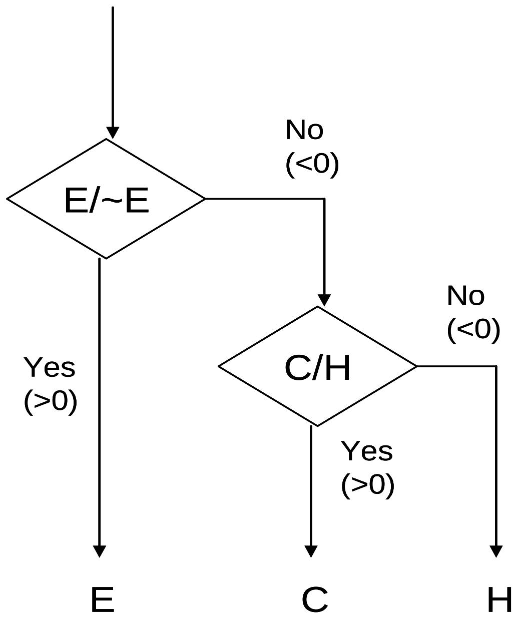
This method is based on one-versus-one binary classifiers (H/~H, E/~E and C/~C) and three one-versus-one classifiers (H/E, E/C and C/H). With these classifiers, three cascade tertiary classifiers, TREE_HEC (H/~H, E/C), TREE_ECH (E/~E, C/H), TREE_CHE (C/~C, H/E) were made. These tree-based classifiers are shown in Figure 4.6



(a) TREE_HEC



(b) TREE_CHE



(c) TREE_ECH

Figure 4-6 Tree based tertiary classifiers

4.6.2. Simple voting tertiary classifier (SVM_VOTE) [3]

In this method, all six binary classifiers are combined by using a simple voting scheme in which the testing sample is predicted to be state i (i is among H, E and C) if the largest number of the six binary classifiers classify it as state i. In case the testing samples have two classifications in each state, it is considered to be a coil.

4.6.3. SVM_MAX_D [3]

In this classifier, the three one-versus-rest classifiers (H/~H, E/~E, C/~C) are combined for handling the multi-class case. And the class of a testing sample (H, E or C) was assigned to the one which presents the largest positive distance from the optimal separating hyperplane (OSH).

For example, if the distance values of the each one-versus-rest classifiers (H/~H, E/~E, C/~C) are -1.7, 1.2 and 2.5 respectively, as negative distance of H/~H binary classifier doesn't give any information for decision, only two positive values (1.2, 2.5) are compared.

Finally, the class for the test sample is assigned to coil because 2.5 is the largest between the two values.

4.6.4. Directed Acyclic Graph (DAG) based tertiary classifier [22]

As shown in Figure 3.8, this classifier is based on three one-versus-one classifiers (H/E, E/C and C/H). Many test results show that one-versus-one classifiers are more accurate than one-versus-rest classifiers due to the fact that the one-versus-rest scheme often need to handle two data sets with very different sizes, i.e. unbalanced training data [23] [24].

In this scheme, if the testing point is predicted to be H (not E) from H/E classifier, then C/H classifier is applied to determine if it is sheet or coil. In Figure 3.8 this DAG scheme is illustrated.

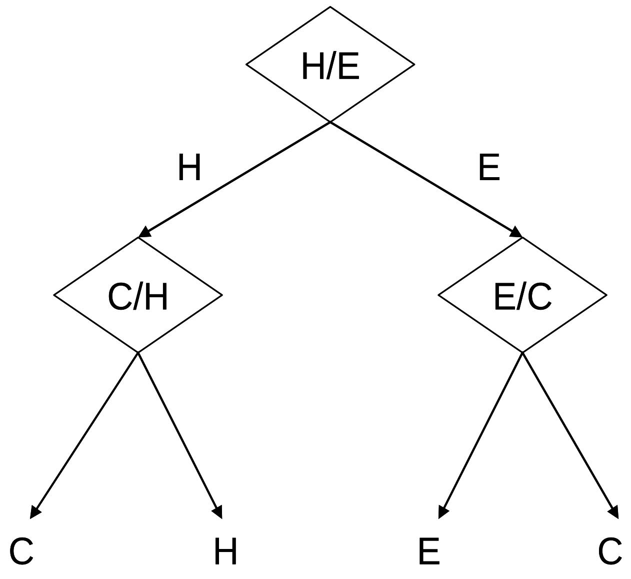


Figure 4-7 DAG scheme

4.6.5. SVM_Representative [6]

This method is similar to the SVM_MAX_D classifier in that the maximum distance is used for the decision. But unlike the SVM_MAX_D classifier, this scheme combines the three one-versus-one binary classifiers (H/E, E/C and C/H) which give more information than one-versus-one binary classifier provides. In one-versus-one classifier, both positive and negative values are meaningful to assign final class but in one-versus-rest classifier, negative value doesn’t provide any specific information for the decision.

In this scheme, no matter what the distance values are positive or negative, the classifier with the absolute maximum distance is chosen as the representative classifier for the final decision of the class.

For example, if the distance values of the each one-versus-one classifiers (H/E, E/C, C/H) are -1.7, 1.2 and -2.5 respectively, the binary classifier with highest absolute value, here C/H classifier, can be chosen for deciding the final class. Once this representative classifier is selected, the final class is assigned based on the value of this classifier. In this example, since the value of C/H classifier shows negative, the final class is assigned as helix.

4.6.6. Multi-class SVM

In multi-class SVM, instead of combining the results of binary classifiers (Fig. 3.8(a)), all classes are considered in one step (Fig. 3.8(b)). Nguyen and Rajapakse [25] applied the multi-class SVM method for secondary structure prediction. The authors examined two multi-class SVM algorithms proposed by Vapnik and Weston [26] [27] and by Crammer and Singer.[28] Even though the authors reported that Vapnik and Weston's multi-class SVM showed better performance than other schemes including combined binary SVM, the accuracy improvement is trivial (less than 0.5 %). Considering the mathematical complexity of the multi-class SVM, it is not clear that this scheme is more suitable for protein secondary structure prediction than the combined binary SVM.

5. NEW SVM-BASED METHODS

5.1. Basic Concepts

5.1.1. Neural Networks

A neural network, also known as a parallel distributed processing network, is a computing paradigm that is loosely modeled after cortical structures of the brain. It consists of interconnected processing elements called nodes or neurons that work together to produce an output function. The output of a neural network relies on the cooperation of the individual neurons within the network to operate. Here neural network is used to form the new tertiary architecture. The neurons in this network are SVM machines that classify the input data into two classes. The two classes are the binary classes that the SVM machine was actually trained for. The architecture is explained in subsequent chapters.

5.1.2. Genetic Algorithms

A genetic algorithm (or GA) is a search technique used in computing to find true or approximate solutions to optimization and search problems. Genetic algorithms are categorized as global search heuristics. Genetic algorithms are a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination).

Genetic algorithms are implemented as a computer simulation in which a population of abstract representations (called chromosomes or the genotype or the genome) of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem evolves toward better solutions. Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

In the tertiary architecture using neural network, genetic algorithm is used to train the neural network. This architecture is termed as 'the Neural Network Architecture'. The weights obtained by training the neural network is then used in testing phase.

5.2. Multiple Windows : New Encoding Scheme

In the case of multiple window scheme which is the new encoding scheme introduced in this study, instead of a single sliding window multiple sliding windows are used. The center element of the middle window becomes the target and all other windows are used as feature values to train and test the SVM. Only the elements/residues inside the window forms the training/testing data, some residues in the middle are skipped. Sliding windows technique is applied to move to the next residue. In this study only windows of equal size is studied. In future windows of different sizes will be studied. When considering windows of different sizes, the window in the middle will have more residues than windows in side.

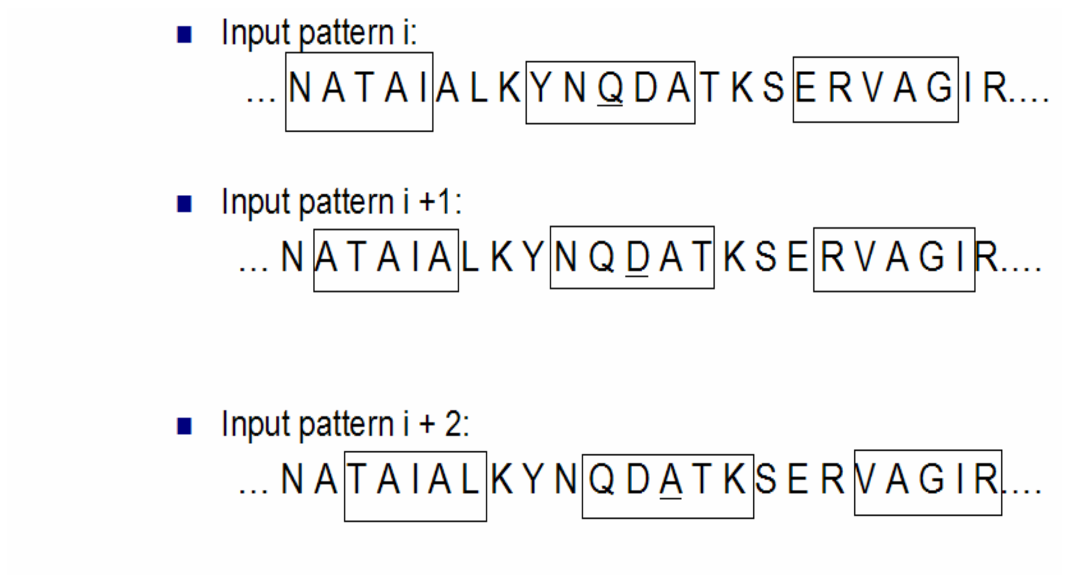

Input pattern i:
... N A T A I A L K Y N Q D A T K S E R V A G I R....
Input pattern i +1:
... N A T A I A L K Y N Q D A T K S E R V A G I R....
Input pattern i + 2:
... N A T A I A L K Y N Q D A T K S E R V A G I R...

Figure 5-1 Multiple Window Scheme

5.3. Genetic Neural Network using SVM Neurons: SVM_GA+NN

The new tertiary classifier proposed in this paper, makes use of both one-versus-one as well as one-versus-rest binary classifiers. The novel architecture makes use of all the six binary classifiers in neural net architecture. The architecture is shown in the Figure 5.2.

In the first phase of the construction of the architecture, the SVM (binary classifiers) are formed to perform their best (i.e. by using optimal window size in the case of single window or optimal slide size and window sizes in the case of multiple windows scheme; and also considering the optimal parameters for the construction of the RBF kernel, as it has been proved by the previous works that RBF kernel has superior performance in the generalization ability and convergence speed). Based on the former studies the binary classifier has an average nearing 80%.

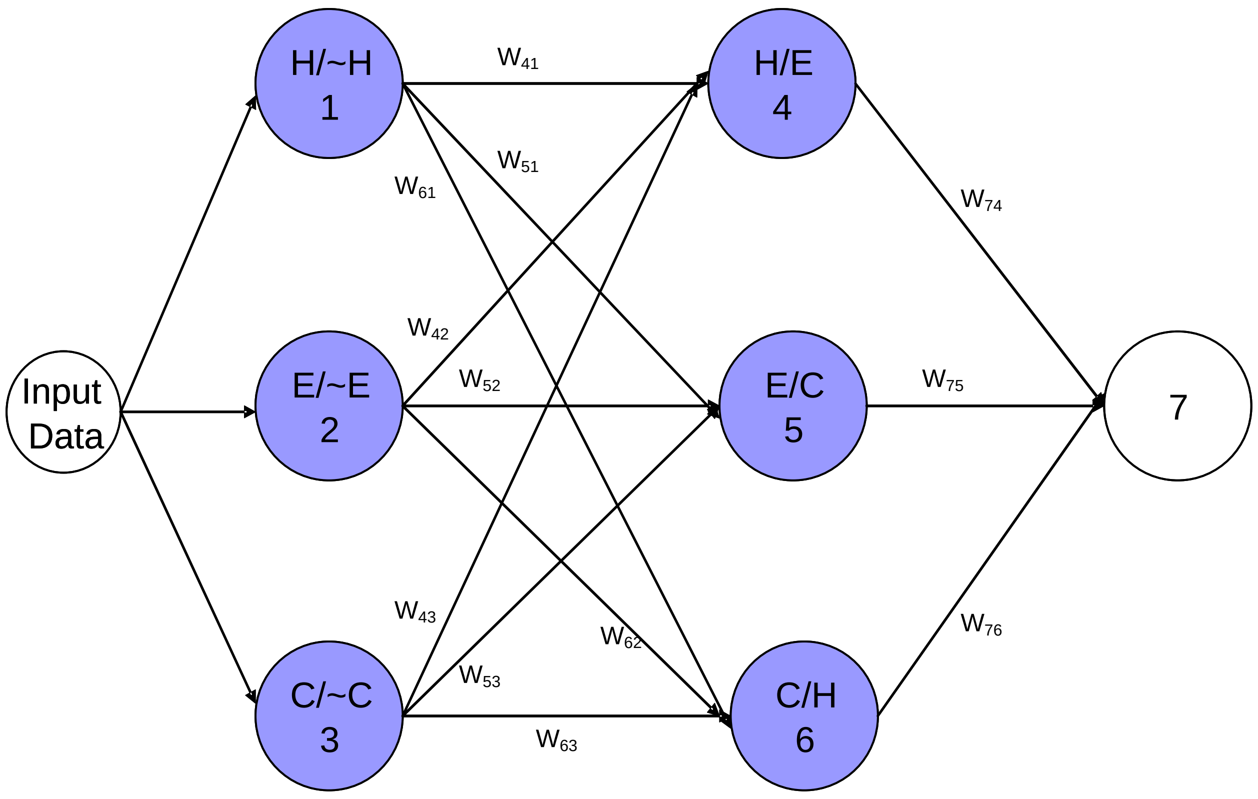


Figure 5-2 Novel Neural Network Using SVM Neurons

The next phase is to use of the new neural network architecture. In this architecture the SVM are used as neurons as shown in Figure 5.2. There are two hidden layers; the output of the first one is the same as the output of the individual SVM.

Outputs of first hidden layer

O_1 = SVM (H/~H)

O_2 = SVM (E/~E)

O_3 = SVM(C/~C)

The output of the second hidden layer considers the output of the first layer as well as the SVM machine stored in that layer. For example the output of the neuron 4 has an SVM binary classifier that positively classifies H and negatively classifies E. the result of this SVM is combined with that of the first layer outputs. This method uses all the outputs of the three one-versus-rest classifier in a single neuron. In the formulations the output of the second hidden layer is formed by adding the output of the SVM sitting inside the neuron with the product of the weight and output of the corresponding neuron (i.e. the neuron which positively classifies the same class as the current neuron) and by subtracting the products of the other two neurons in the first layer.

The output of the second layer are calculated as

$O_4 = SVM\ (H/E) + W_{41} O_1 - W_{42} O_2 - W_{43} O_3$

In the above formula we add the values of the SVM that positively classify the same class and subtract those that positively classify other classes. Here W_{41} means weight between neuron 1 and neuron 4. Similarly other weight responds to output , input naming pattern. (See Figure 5.2).

Similarly outputs of other two neurons in the second hidden layer are calculated as

$O_5 = SVM\ (E/C) + W_{52} O_2 - W_{51} O_1 - W_{53} O_3$

$O_6 = SVM\ (H/E) + W_{63} O_3 - W_{62} O_2 - W_{61} O_1$

The final output layer does not have any SVM embedded in it. It calculates its results based on maximum of the three outputs of second hidden layer. There is only one neuron in this layer. The final output is one among the three classes (H, E or C), which ever neuron produces the maximum output after multiplying it with appropriate weight with the second hidden layer output is considered as final output.

So the output of the third layer is as follows

```
If [Max ($W_{71} O_4$, $W_{75} O_5$, $W_{76} O_6$) = $W_{71} O_4$)]
    Then
        $O_7 = H$
Else If [Max ($W_{74} O_4$, $W_{75} O_5$, $W_{76} O_6 = W_{75} O_5$) ]
    Then
        $O_7 = E$
    Else
        $O_7 = C$
```

For optimizing the weights Genetic Algorithm is used. The weight range was selected to be between 0 and 1 so that the architecture performs to its full potential.

5.4. SVM_Complete: New Tertiary Classifier of this study

In this method all the six binary classifiers are used to form the tertiary classifier. In Hu's scheme, no matter what the distance values are positive or negative, the classifier with the absolute maximum distance is chosen as the representative classifier for the final decision of the class. In this paper, we consider that fact that among the three one-versus-one classifier, two classifier try to identify the same class, for example H/E and C/H tries to classify H (only difference is in H/E H is the positive class and in C/H H is the negative class). So we add up the values of one-versus-one classifier which classifies the same class. Then we also add the value of one-versus-rest classifier, to sum up the total strength of the specific class.

For example, for calculating the strength of H, we have to ..

Step 1: Check if SVM (H/E) is positive, if true

H = absolute value of SVM (H/E)

Step2: Check if SVM (C/H) is negative, if true

H = H + absolute value of SVM (C/H)

Step 3: Add one-versus-rest prediction value

H = H + value of SVM (H/~H). *

* Note here we add the actual value not absolute, since we want to determine H's total strength.

Similarly strength of E and C are calculated and final result is produced depending upon which class has the highest value.

Here SVM (H/E) means the exact output the support vector machine gives after classifying the given data.

6. SIMULATION RESULTS

6.1. Accuracy Measure of SVM

There are several standard evaluation methods of secondary structure prediction. Among them, Q_3, Matthew's Correlation Coefficient and Segment Overlap Measure (SOV) are widely used assessing methods.

6.1.1. Q3

Q_3 is one of the most commonly used performance measures in the protein secondary structure prediction and it refers to the three-state overall percentage of correctly predicted residues. This measure is defined as,

$$Q_3 = \frac{\sum_{i \in \{H,E,C\}} \#of\ residues\ correctly\ predicted_i}{\sum_{i \in \{H,E,C\}} \#of\ residues\ in\ class\ i} \times 100 \qquad (6.1)$$

Based on the above equation, the per-residue accuracy for each type of secondary structure (Q_H, Q_E, Q_C) can be obtained as:

$$Q_I = \frac{\#of\ residues\ correctly\ predicted\ in\ state\ I}{\#of\ residues\ in\ state\ I} \times 100 \qquad (6.2)$$

$I \in \{H, E, C\}$

6.1.2. Matthew's Correlation Coefficient, C_i

Matthew's Correlation Coefficient is another measure used in the protein secondary structure prediction and it shows how closely the prediction is correlated with the results. [4] The Matthew's correlation coefficient is given by

$$Ci = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FN)(TP + FP)(TN + FP)(TN + FN)}} \tag{6.3}$$

$i \in$ {H, E, C}

Where, *TP, FP, FN, TN* are the number of true positives, false positives, false negatives and true negatives for class *i* respectively and for clarity, the dependency on *i* has been dropped on the right hand side. This coefficient value falls on the range between -1 and 1, with 1 showing complete agreement, -1 complete disagreement and 0 showing that the prediction was uncorrelated with the results.

6.1.3. Segment Overlap Measure (SOV)

Segment Overlap Measure was developed by Rost et al [29] and modified by Zemla et al [30] to evaluate the quality of a prediction in a more realistic manner. While all the previous measures are general statistics that can be used in any classification problem, SOV is the specially designed measure for secondary structure prediction. In the secondary structure prediction, it is important that the continuous structural elements are predicted to exist [4]. Even though Q_3 shows high accuracy value, if a continuous element is not predicted as existing continuously, this cannot be a good prediction. By including this knowledge, SOV became a measure suitable for evaluation of the secondary structure segment rather than individual residues. SOV is calculated as: [30]

$$\text{SOV} = \frac{1}{N}\sum_{i\in\{H,E,C\}}\sum_{s(i)}\left[\frac{\min ov(s_1,s_2)+\delta(s_1,s_2)}{\max ov(s_1,s_2)}\times len(s_1)\right]\times 100 \tag{6.4}$$

Where,

N is the normalization value and a sum of *N*(*i*) over all three states,

$$N = \sum_{i\in\{H,E,C\}} N(i) \tag{6.5}$$

$$N(i) = \sum_{s(i)} len(s_1) + \sum_{s'(i)} len(s_1) \tag{6.6}$$

S(*i*) is the set of all overlapping pairs of segments(s_1, s_2) in state *i,*

$S(i) = \{(s_1, s_2) : s_1 \cap s_2 \neq \varnothing, \textit{ in state } i\}$

S' (*i*) is the set of segments (s_1, s_2) in state *i* for which there is no overlapping,

$S'(i) = \{s_1 : \forall s_2,\ s_1 \cap s_2 = \varnothing, \textit{ in state } i\}$

len(s_1) is the number of residues in segment s_1,

$len(s_1) = e(s_1) - b(s_1) + 1$

$b(s_1)$ is the position at which segment s_1 begins and $e(s_1)$ is the position at which segment s_1 ends

minov(s_1, s_2) is the length of the actual overlap:

$minov(s_1, s_2) = min(e(s_1), e(s_2)) - max(b(s_1), b(s_2)) + 1$

maxov(s_1, s_2) is the total extent of the segment expressed as:

$$maxov(s_1, s_2) = max(e(s_1), e(s_2)) - min(b(s_1), b(s_2)) + 1$$

and $\delta(s_1, s_2)$ is given as:

$$\delta(s_1, s_2) = min\{(maxov(s_1, s_2) - minov(s_1, s_2)), minov(s_1, s_2), int(len(s_1)/2), int(len(s_2)/2) \}$$

The quality of the matching of each segment pair is taken as a ratio of the overlap of the two segments *minov*(s_1, s_2) and the total extent of that pair *maxov*(s_1, s_2). The definition of δ and the normalization factor N is different between SOV94 [29] and SOV99 [30].

6.2. Single Window vs. Multiple Windows

The first main research of this study is the new technique of using multiple windows instead of the single window technique used in Hu's research [6] and former studies. The comparison of the two techniques revealed single window scheme not to be good in all cases. For window 15 the simulation results showed the multiple windows to be better than single window for all the six binary classifiers. The results are shown in the table 6.1

Table 6-1 Comparing Single Window and Multiple Windows

Binary Classifier	**Multiple Window**	**Single Window**
H/~H	73.59%	73.52%
E/~E	78.39%	78.39%
C/~C	69.69%	69.62%
H/E	72.94%	72.33%
E/C	75.9%	75.59%
C/H	71.93	71.74
Average	73.74%	73.53%

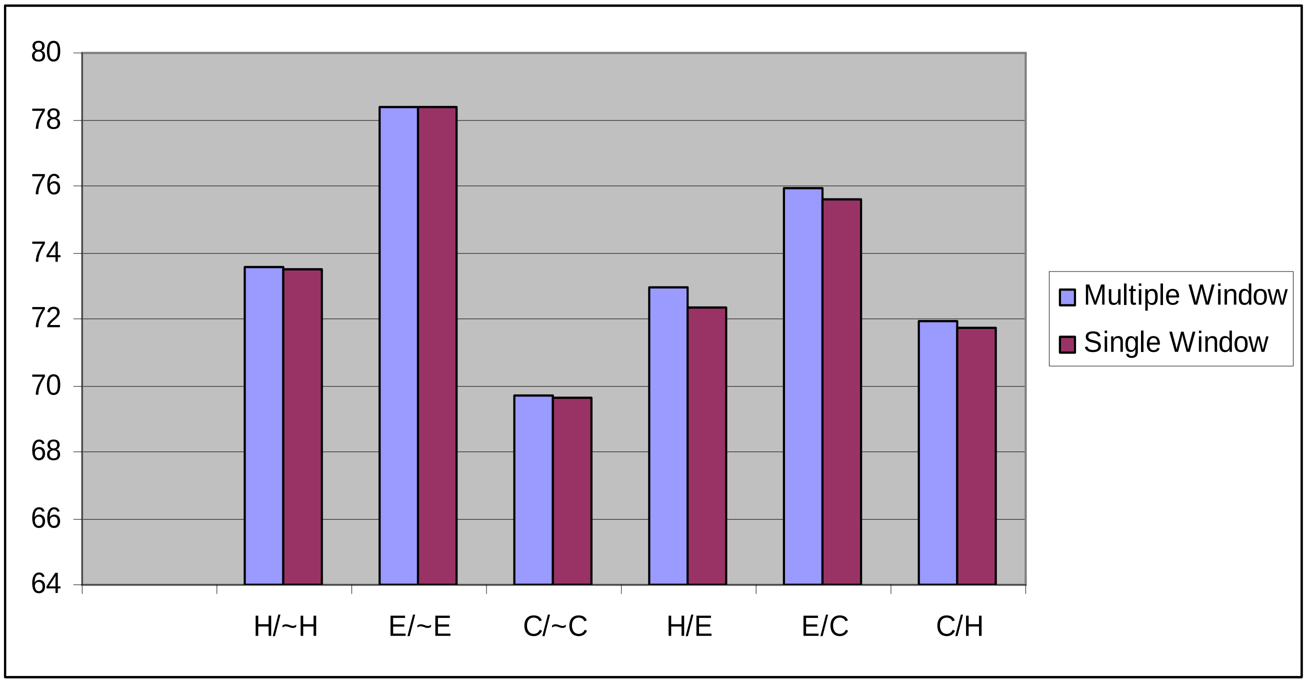


Figure 6-1 Comparing Single Window and Multiple Windows.

In the above case the single window is of length 15 and in the multiple windows case, 3 windows each of size 5 with gaps between the windows are used. In both the cases RBF kernel is used with the same parameter values (gamma γ and cost co-efficient C). Another kind of simulation was done in which a single window of size 21 was compared with 3 windows, each of size 5 and a gap of 3 residues (gap means these three residue was not considered to form the data for SVM) between the windows. The results of this simulation are shown in Table 6.2. Its shows that single window not be good in all cases and multiple windows had less information to process as it had only 15 residues to consider where as single window 21 residues in each set. Considering all the points multiple windows still showed some scope of performance. This study was conducted to see if single is solely the best method to go with protein secondary structure prediction, empirically there is scope for other methods too.

Table 6-2 Simulation II: Single Window vs. Multiple Windows

Binary Classifier	**Accuracy of Multiple Windows**	**Accuracy of Single Window**
H/~H	72.37%	74.67%
E/~E	78.41%	78.34%
C/~C	70.00%	69.63%
H/E	72.24%	73.7%
E/C	75.45%	74.3%
C/H	71.43%	72.9%
Average	73.32%	73.92%

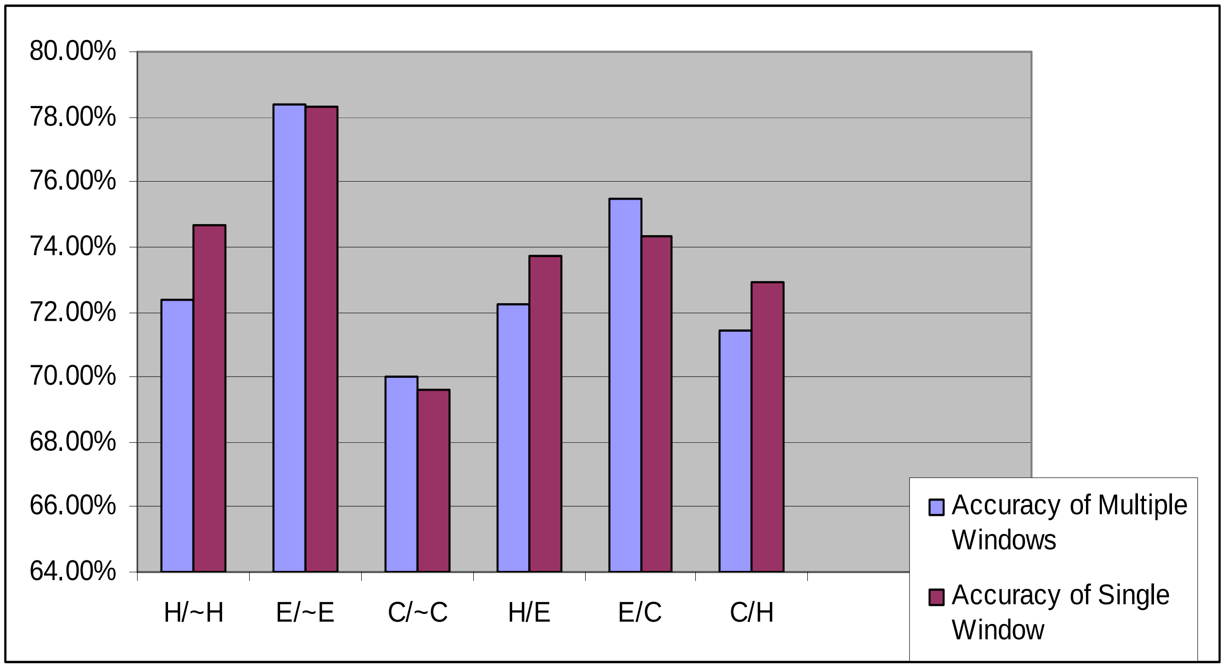


Figure 6-2 Simulation II: Single Window vs. Multiple Windows.

The optimal window length and other optimal values of the parameters are selected to be the same as those used in previous studies. As the previous studies have already run simulations and have obtained the optimal values for all the parameters, further research is avoided.

6.3. Tertiary Classifiers

The two new tertiary classifiers were compared with other tertiary architectures of former studies. The neural network architecture which uses genetic algorithm for optimization was seen to perform better in most cases. The 7 fold test cases have been performed (included in Appendix) for a valid comparison of the new tertiary classifiers with that of the Hu [6] contributed classifiers. The accuracy percentage of the new methods is compared with that of the other methods. The accuracy level of the tertiary classifier is important from research point of view, as the main objective in this study is to accurately determine the secondary structure of the protein sequence. The Table 6.3 gives the accuracy level of all the methods [6] and also the accuracy levels of the neural network architecture as well the new SVM_Complete (Second tertiary classifier of this research). As shown in the table SVM_Complete and the neural network architecture is better than other available methods.

The are many researches that show accuracy greater than 75 % but all this studies use PSSM (Position Specific Scoring Matrix) as their encoding scheme for binary classifier. The reason for higher tertiary classifier accuracy is due to the fact that their binary classifiers have over 85 % of prediction accuracy.

6.3.1. Single Window Encoding Scheme

First the accuracy levels of single window encoding scheme was compared with different former methods. As seen in table 6.3 the average accuracy of the neural network architecture and that of SVM-Complete (a new classifier of this study) is better than other available methods. Closely analyzing the accuracy levels, it is recorded that Neural Network of SVM has performed equally well in all 3 cases (H, E and C), when compared to other methods that have very high Q_C accuracy and have very low Q_E accuracy. The 'Neural Network of SVM' still has scope of improvement as it is a neural network technique which is optimized using Genetic Algorithms, its potential can be further increased. The table 6.3 is the accuracy level for window of size 15.

Table 6-3 Accuracy of tertiary Classifiers on the RS 126 data set. Combined results of 7-fold cross validation are shown.

Tertiary Classifier	**Q_3(%)**	**Q_H(%)**	**Q_E(%)**	**Q_C(%)**
TREE_HEC	63.2	51.0	45.2	79.9
TREE_ECH	62.3	62.4	26.2	79.0
TREE_CHE	61.2	64.8	47.3	65.2
SVM_VOTE	62.0	73.5	34.7	65
SVM_MAX_D	63.2	61.0	40.1	75.5
DAG	63.2	59.2	41.6	76.0
SVM_REPRESNT.	63.2	70.6	35.4	70.5
SVM_Complete	66.7	64.0	40.8	80.3
SVM_GA+NN	66.1	68.3	49.8	72.1

* The table is adopted from Hu's research [6].

SVM_VOTE classifier which is introduces by Hua [3] is shown to have Q_3 accuracy of 68.3 % in Hu's paper [6], the window size for which this accuracy was obtained is unknown. The results in the table are obtained after 7-fold cross validation for window of size 15. The accuracies are compared with other classifiers that use single window encoding scheme. In the table 6.3 accuracy levels of SVM_Represnt. and SVM_VOTE are obtained by simulation after 7-fold cross validation. All other former classifiers accuracies are adopted from [6].

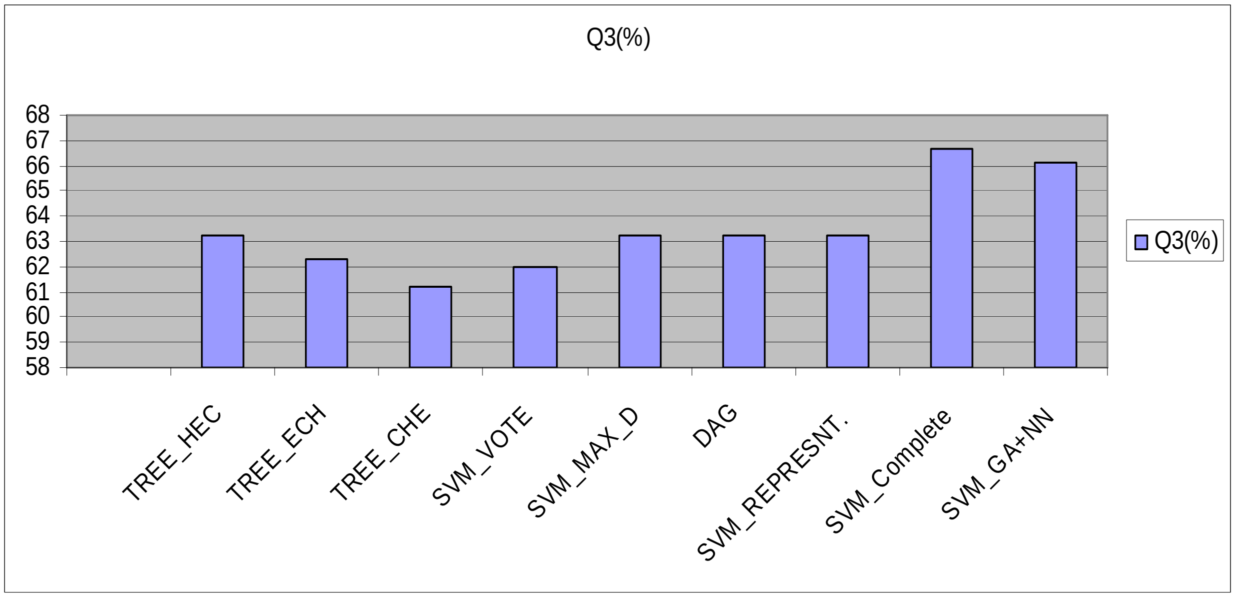


Figure 6-3 Accuracy of tertiary classifiers on the RS 126 dataset

For giving more completeness to the argument of which tertiary classifier is best, 7 fold testing is performed for different window sizes. This 7 – fold testing results are shown in Table 6.4, 6.5 and 6.6 for window sizes 15, 13 and 11. The outcome of these results shows that neural network and SVM_Complete is better than SVM_Represnt., which claims to have the best performance using PSSM profile. These simulation results are the outcome of using the same SVM machines in all the cases.

Table 6-4 Q_3 (%) after 7-fold Cross Validation for Window of size 15.

Test Case	**SVM_GA+NN**	**SVM_Complete**	**SVM_Represnt.**
1	65.08	63.97	60.10
2	65.44	65.41	63.55
3	64.45	65.89	63.19
4	67.93	68.23	65.38
5	67.75	69.04	63.79
6	66.04	67.57	62.92
7	66.01	66.74	63.11
Average	66.10	66.70	63.15

Table 6-5 Q_3 (%) after 7-fold Cross Validation for Window of size 13.

Test Case	**SVM_GA+NN**	**SVM_Complete**	**SVM_Represnt.**
1	55.99	55.57	54.37
2	64.59	66.31	63.69
3	65.56	67.89	66.69
4	68.14	69.10	66.84
5	69.39	70.50	68.74
6	54.38	54.47	53.40
7	63.99	64.77	63.30
Average	63.15	64.09	62.43

Table 6-6 Q_3 (%) after 7-fold Cross Validation for Window of size 11.

Test Case	SVM_GA+NN	SVM_Complete	SVM_Represnt.
1	51.58	51.49	51.61
2	57.97	57.40	55.60
3	60.52	60.61	59.83
4	56.92	56.95	55.27
5	59.06	57.73	56.83
6	57.43	57.61	56.17
7	55.87	55.93	56.17
Average	57.05	56.82	55.93

In above tables, the SVM_Represnt. classifier [6] is compared with the two new classifiers of this study and in every case the classifier SVM_Complete has the best average. Though neural network of SVM architecture is not the best, it is arguable that it has not been made to perform to its full potential. Since it uses Genetic Algorithm, its performance can change with the number of generations, population size and also the range of weights. These factors determine its prediction accuracy to some extent.

6.3.2. Multiple Windows Encoding Scheme

The same tertiary classifiers were demonstrated with binary classifiers using multiple window scheme. This resulted in increase in total accuracy level, which is expected as the binary classifiers formed using multiple window scheme are better when compared to single window encoding scheme. The binary classifier used is constructed using three consecutive windows each of size 5 with gaps between the first and second window as well as between second window and third window. The results of this simulations are shown in Table 6.7.

Table 6-7 Accuracy of Tertiary Classifier Using Multiple Windows Scheme

Tertiary Classifier	Q3(%)	QH(%)	QE(%)	QC(%)
SVM_VOTE	62.6	78.9	39.7	62.4
SVM_REPRESNT.	64.8	72.1	41.8	72.0
SVM_Complete	68.4	69.1	45.0	78.8
SVM_GA+NN	68.0	73.5	52.3	71.7

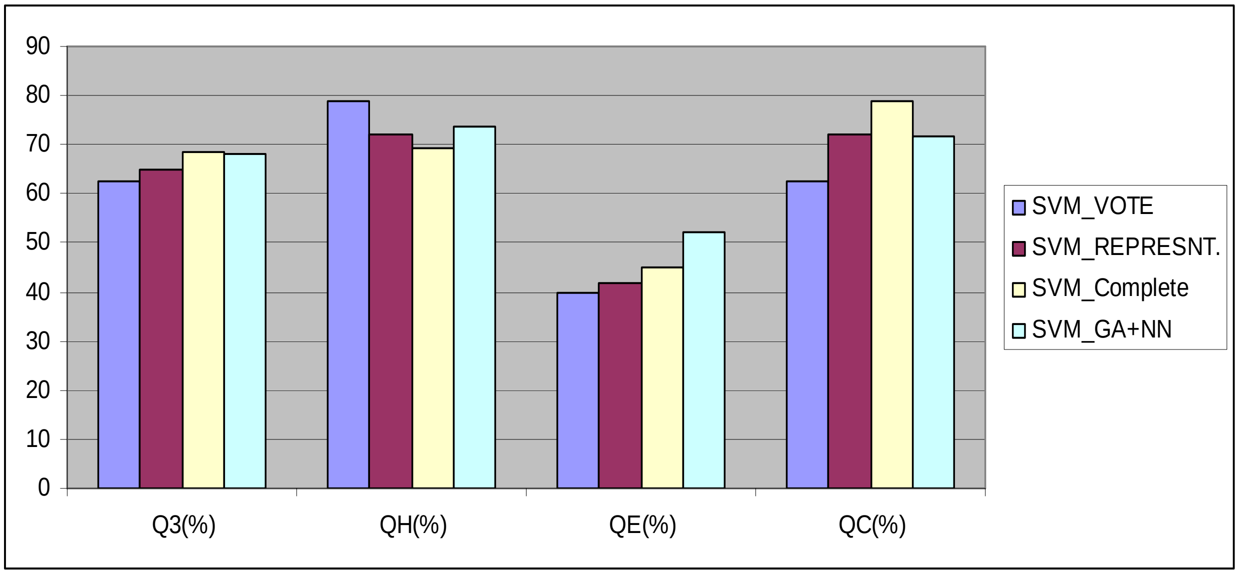


Figure 6-4 Showing Q3, QH, QE and QC % of Different Classifiers Using Multiple Windows Scheme

The accuracy levels for individual secondary structure (H/E/C) is shown to emphasize how each tertiary classifier classifies the secondary structures. For example SVM_GA+NN has the highest accuracy level while predicting sheet (E) structure, when compared with other tertiary classifiers.

To compare all the results, Table 6.8 is shown. This table combines Table 6.3 and Table 6.7. This table shows that among all the tertiary classifiers SVM_Complete (tertiary classifier of this study) is the best.

Table 6-8 Accuracy levels of Tertiary Classifiers

Tertiary Classifier	**Q_3(%)**	**Q_H(%)**	**Q_E(%)**	**Q_C(%)**
TREE_HEC	63.2	51.0	45.2	79.9
TREE_ECH	62.3	62.4	26.2	79.0
TREE_CHE	61.2	64.8	47.3	65.2
SVM_VOTE	62.0	73.5	34.7	65.0
SVM_MAX_D	63.2	61.0	40.1	75.5
DAG	63.2	59.2	41.6	76.0
SVM_REPRESNT.	63.2	70.6	35.4	70.5
SVM_Complete	66.7	64.0	40.8	80.3
SVM_GA+NN	66.1	68.3	49.8	72.1
SVM_VOTE*	62.6	78.9	39.7	62.4
SVM_REPRESNT.*	64.8	72.08	41.84	71.99
SVM_Complete*	68.4	69.09	44.98	78.78
SVM_GA+NN*	68.0	73.44	52.38	70.68

* Results Obtained using Multiple Windows (size 15) encoded binary classifiers.

6.4. Different Encoding Scheme

In Hua and Sun [3] study, they adopted frequency matrix (SVMfreq) with multiple sequence alignments and the SVMpsi is the scheme from Kim and Park [22] which applied the PSSM obtained by PSI-BLAST searches. But the encoding scheme used in our method is different, it is the combination of orthogonal and BLOSUM62 matrix adopted from Hu's research [6]. The accuracy level of this study is about 5 – 14 % lower than SVMpsi and it is about 0.4-5%A lower than SVMfreq with RS126 data set. This fact infers that the encoding scheme of SVMpsi or SVMfreq might be a better choice for the performance of binary classifier. [6]. PSSM matrix should be used as an encoding scheme in the future study and the use of the same tertiary classifier (Neural Network of SVM and SVM-Complete) would result in better prediction accuracy.

7. CONCLUSIONS AND FUTURE WORKS

7.1. Conclusions

The conclusion of this study is separated into two categories, the first one deal with the comparison of single window encoding scheme with the multiple window encoding scheme. The second category deals with tertiary classifier's accuracy level assessment.

In this study, first the accuracy of single window scheme is compared with that of the multiple windows scheme. After many demonstrations, it is now established that single window scheme is not the only best method to encode. Multiple windows scheme performed better in some cases where the data given to the learning machine (SVM) was less informative than that given in single window scheme. When both were encoded with equal amount of information, multiple window schemes' performance is better than single window scheme in every case.

All the tertiary classifiers discussed in this study have less accuracy when directly compared with new tertiary classifiers introduces in this study. Though the study is not better when compared directly to the claimed accuracy levels of the former methods, the encoding scheme of binary classifiers used in those methods is different and better than the one used in this study. But when the same architecture was tested with the encoding method of this study, clearly the two new architecture of this study performed much better than most of the other existing methods.

The SVM is used as the training engine in this study, which has proved to be the best method to be used for secondary structure prediction by all former studies.

7.2. Future Works

The future work of both the categories primarily deals with using different encoding schemes, which will increase the results of both binary as well tertiary classifier's accuracy levels. More concrete case can be developed if other datasets like CB513 etc. is used to prove the supremacy of these new methods over other contemporary techniques.

Though the best encoding scheme (PSSM) was not tested to establish this results, still we can conclude that single window scheme not be the sole method for forming the training data for SVM and to accurately predict the secondary structure of proteins. As a future work, multiple windows encoding scheme should be tested using PSSM and frequency profiling [3] as encoding scheme to form the data for Support vector machines, as this methods form more accurate binary classifiers when compared with the encoding scheme of BLOSUM62 and orthogonal method which was used in this method. Multiple windows scheme is new method proposed in this study; its total potential is still not established. Multiple windows with different window sizes will be considered for future studies.

In the future work, PSS matrix will be used to form the training data from RS 126 dataset, to train and test SVM. Also frequency profiling technique used by Hua and Sun [3] will also be tested to see if multiple windows scheme is better than single window scheme. After forming the best binary classifiers, the new tertiary classifiers will be tested to prove that their performance is best among all the current research methods.

REFERENCES

[1] Rost, B. and Sander, C. *Improved prediction of protein secondary structure by use of sequence profile and neural networks*. Proc Natl Acad Sci U S A 90, 7558-62 (1993).

[2] Chandonia, J.M., and Karplus, M. *New Method for accuracy prediction of protein secondary structure*. Proteins 35, 293-306 (1999).

[3] Hua, S. and Sun, Z. *A Novel Method of Protein Secondary Structure Prediction with High Segment Overlap Measure: Support Vector Machine Approach*. J. Mol. Biol. 308, 397-407 (2001).

[4] Casbon, J. *Protein Secondary Structure Prediction with Support Vector Machines*. (2002).

[5] Jones D.T. *Protein Secondary Structure Prediction Based on Position-Specific-Scoring Matrices*. J. Mol. Biol., 292,195-202, (1999).

[6] Hu, H. Yi, P. *Improved Secondary Structure Prediction Using Support Vector Machines with a New Encoding Scheme and an Advanced Tertiary Classifier.* IEEE – Transaction on Nanobioscience, Vol. 3, No. 4 (2004)

[7] V.Vapnik and C. Corter. *Support vector networks* . Machine Learning. Vol. 20 pp. 273-293, 1995.

[8] Berg, J.M., Thorsson, V, and Baker, D. HMMSTR: " *a hidden Markov model for local sequence-structure correlations in protein.* J. Mol. Biol. 301, 173-90 (2000).

[9] Kabsch, W. and Sander, C. *Dictionary of Protein Secondary Structure : Pattern recognition of hydrogen-bonded and geometrical feature.* Biopolymers 22, 2577-2637 (1983).

[10] Chou, P.Y. "*Prediction of Protein Structure and the Principles of Protein Conformation",* New York, Plenum Press (1989).

[11] Garnier, J., Osguthorpe, D.J., and Robson, B. *Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins*. J. Mol. Biol 120, 97-120 (1978).

[12] Cuff, J.A. & Barton, G.J Evaluation and improvement of multiple sequence methods for protein secondary structure prediction, *Proteins: Structure, Function, and Genetics*, 34, pp. 508-519. (1999)

[13] Przybylski, D. and Rost, B. *Alignments grow, secondary structure prediction improves*. Proteins 46, 197-205 (2002).

[14] Salamov, A.A. & Solovyev, V.V. *Protein secondary structure prediction using local alignments*, *J. Mol. Biol*, 268, pp. 31-36. (1997)

[15] Frishman, D. & Argos, P. 75% accuracy in Protein Secondary Structure Prediction, *Proteins*, 27, pp. 329-335.(1997)

[16] Mount, D.W. *Bioinformatics: Sequence and Genome Analysis. Cold Spring Harbor Laboratory Press*. New York (2001).

[17] Bystroff, C., Thorsson, V. & Baker, D. HMMSTR: *A hidden Markov model for local sequence-structure correlations in proteins*, *J Mol Biol*, 301, pp. 173-190. (2000)

[18] Christianini, N. and Shawe-Taylor, J. *An introduction to Support Vector Machines* Cambridge University Press (2000).

[19] Burges, C.J.C. *A Tutorial on Support Vector Machines for Pattern Recognition* (1998). http://www.kernel-machines.org/papers/Burges98.ps.gz

[20] Joachims, T. SVM light. http://www.cs.cornell.edu/People/tj/svm_light/ (2002).

[21] Henikoff, S. and Henikoff, J.G. *Amino acid substitution matrices from protein blocks*. PNAS 89, 10915-10919 (1992).

[22] Kim, H. and Park, HProtein Secondary Structure Prediction Based on an Improved Support Vector Machines Approach. *Protein Eng*, 16, 553-560. . (2003)

[23] Chang, C.C. & Lin, C.J. LIBSVM : a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm (2001)

[24] Heiler, M. *Optimization Criteria and Learning Algorithms of Large Margin Classifiers*, University of Mannheim. (2002)

[25] Nguyen, M.N. & Rajapakse, J.C. Multi-Class Support Vector Machines for Protein Secondary Structure Prediction, *Genome Informatics*, 14, pp. 218-227.(2003)

[26] Weston, J. & Watkins, C. Multi-class support vector machines in: Verleysen, M. (Ed) *Proceedings of ESANN99*, Brussels, D. Facto Press.(1999)

[27] Vapnik, V. *Statistical Learning Theory*, New York, Wiley and Sons, Inc.(1998)

[28] Crammer, K. & Singer, Y. "On the learnability and design of output codes for multi-class problems, *Computational Learning Theory*, pp. 35-46. (2000)

[29] Rost, B., Sander, C. & Schneider, R. Redefining the goals of protein secondary structure prediction., *J. Mol. Biol.*, 235, pp. 13-26. (1994)

[30] Zemla, A., Venclovas, C., Fidelis, K. & Rost, B. A modified definition of sov, a segment-based measure for protein secondary prediction assessment, *Proteins: Structure, Function, and Genetics*, 34, pp. 220-223. (1999)

[31] Source http://www.medchem.umu.se/PDF/Jurgen/proteinstructure.pdf

[32] Reported from ref. http://www.pasteur.fr/recherche/united/neubiomol/sectrpr.html

APPENDIX

BLOSUM62 MATRIX

A	**R**	**N**	**D**	**C**	**Q**	**E**	**G**	**H**	**I**	**L**	**K**	**M**	**F**	**P**	**S**	**T**	**W**	**Y**	**V**	*****	
4	-1	-2	-2	0	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1	0	-3	-2	0	-4	**A**
	5	0	-2	-3	1	0	-2	0	-3	-2	2	-1	-3	-2	-1	-1	-3	-2	-3	-4	**R**
		6	1	-3	0	0	0	1	-3	-3	0	-2	-3	-2	1	0	-4	-2	-3	-4	**N**
			6	-3	0	2	-1	-1	-3	-4	-1	-3	-3	-1	0	-1	-4	-3	-3	-4	**D**
				9	-3	-4	-3	-3	-1	-1	-3	-1	-2	-3	-1	-1	-2	-2	-1	-4	**C**
					5	2	-2	0	-3	-2	1	0	-3	-1	0	-1	-2	-1	-2	-4	**Q**
						5	-2	0	-3	-3	1	-2	-3	-1	0	-1	-3	-2	-2	-4	**E**
							6	-2	-4	-4	-2	-3	-3	-2	0	-2	-2	-3	-3	-4	**G**
								8	-3	-3	-1	-2	-1	-2	-1	-2	-2	2	-3	-4	**H**
									4	2	-3	1	0	-3	-2	-1	-3	-1	3	-4	**I**
										4	-2	2	0	-3	-2	-1	-2	-1	1	-4	**L**
											5	-1	-3	-1	0	-1	-3	-2	-2	-4	**K**
												5	0	-2	-1	-1	-1	-1	1	-4	**M**
													6	-4	-2	-2	1	3	-1	-4	**F**
														7	-1	-1	-4	-3	-2	-4	**P**
															4	1	-3	-2	-2	-4	**S**
																5	-2	-2	0	-4	**T**
																	11	2	-3	-4	**W**
																		7	-1	-4	**Y**
																			4	-4	**V**
																				1	*****