國 立 交 通 大 學

電 機 與 控 制 工 程 研 究 所

碩 士 論 文

使用二級式 QuickRBF 及 Fuzzy ARTMAP 演算法於蛋白質二級結構預測

Two-Stage QuickRBF and Fuzzy ARTMAP to Protein Secondary Structure Prediction

研 究 生：林 彥 宏

指 導 教 授：張 志 永

中 華 民 國 九十五 年 六 月

使用二級式 QuickRBF 及 Fuzzy ARTMAP 演算法於蛋白質二級結構預測

Two-Stage QuickRBF and Fuzzy ARTMAP to Protein Secondary Structure Prediction

學　　生：林彥宏　　Student : Yen-Hung Lin

指導教授：張志永　　Advisor : Jyh-Yeong Chang

國立交通大學

電機與控制工程學系

碩士論文

A Thesis

Submitted to Department of Electrical and Control Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of Master in

Electrical and Control Engineering

June 2006

Hsinchu, Taiwan, Republic of China

中 華 民 國 九 十 五 年 六 月

使用二級式 QuickRBF 及 Fuzzy ARTMAP 演算法於蛋白質二級結構預測

學生：林彥宏　　指導教授：張志永博士

國立交通大學電機與控制工程研究所

摘要

隨著人類基因定序及許多基因定序計畫陸續完成，序列的資料量將大幅成長，有效地分析這些序列更顯得重要了。基於物運作的原則（Central Dogma），蛋白質的功能與結構遂成為相當重要的研究議題，而目前在蛋白質相關問題的解決上，科學家都是利用X光繞射以及核磁共振 (NMR) 來取得實驗結果。這些方法雖然正確率高，但是相對地所要花費的時間及成本是相當高的。因此，研究人員便利用電腦科學來協助解決這些問題，相信是能夠有效降低實驗成本的。由於要了解完整蛋白質的功能必需從三級結構著手，但直接從蛋白質序列去預測它的三級結構是非常困難的議題，因此一個間接且有助益的方式，便是預測其二級結構。過去的研究中，學者們通常將蛋白質二級結構分成三種類別，分別是螺旋體（helix）、摺疊（sheet）、其他部份歸類為迴圈（loop）。因此我們可以將蛋白質二級結構預測視為一個普遍的分類問題

本篇論文中，我們提出利用二層快速徑向基底函數網路分類器（Quick Radial Basis Function）來預測蛋白質二級結構。快速徑向基底函數網路分類器能夠迅速地建構分類器的模型，其預測準確性更是不亞於目前廣受歡迎的機器學習演算法支持向量機（Support Vector Machine）。最後，將各層分類之結果合併，而有效的提高預測之準確度。在本研究中，我們使用著名的 RS126 資料集，以及 PSI-BLAST 所產生的 PSSM，所達到的最佳準確率為76.7%。

Two-Stage QuickRBF and Fuzzy ARTMAP to Protein Secondary Structure Prediction

STUDENT: YEN-HUNG LIN ADVISOR: JYH-YEONG CHANG

Institute of Electrical and Control Engineering

National Chiao-Tung University

ABSTRACT

The majority of human coding regions have been sequenced and several genome sequencing projects have been completed. With the growth of large-scale sequencing data, an efficient approach to analyze protein is more important since protein function and structures are crucial issues in bioinformatics. Nowadays, scientists use X-ray diffraction or nuclear magnetic resonance (NMR) to solve the protein structure problems. Even though chemical experiments can achieve high accuracy, they in the mean time incur high cost and long time to solve the protein problems. Hence, computational tools are then applied thereto and considered as promising ways which not only reduce the time and the cost but also maintain reliable predictive results. The protein secondary prediction (PSS) is an intermediate but useful step for the three-dimensional (tertiary) structure prediction. In the previous work, researchers always focused on classifying three states of protein secondary structure: helix, strand and coil classes. It is a common classification problem for the prediction of protein secondary structure.

In this thesis, a high-performance method was developed for protein secondary structure prediction based on the dual-layer QuickRBF technology that has been

successfully applied in solving problems in the field of bioinformatics. The QuickRBF is capable of delivering the same level of performance as the state of art approach, SVM, while having execution efficiency during the phase to construct the classifier. The performance was further improved by combining PSSM profiles with the QuickRBF analysis where the PSSMs were generated from PSI-BLAST profiles, which contain important evolution information. The final prediction results were generated from the first fusion method. We report a maximum prediction accuracy of 76.7% on the famous RS126 dataset based on the PSI-BLAST profiles.

ACKNOWLEDGEMENT

I would like to express my sincere appreciation to my advisor, Dr. Jyh-Yeong Chang. Without his patient guidance and inspiration during the two years, it is impossible for me to overcome the obstacles and complete the thesis. In addition, I am thankful to all my lab members for their discussion and suggestion.

Finally, I would like to express my deepest gratitude to my parents. Without their fully support and encouragement, I could not go through these two years.

Content

List of Figures

List of Tables

Chapter 1. Introduction

1.1 Motivation and the Background of This Research

The number of known proteins and its structure has been increased in recent years. Since protein applications are more widely used, there will be a lot of problems to be solved. Nowadays, scientists use X-ray diffraction or nuclear magnetic resonance (NMR) to solve the protein structure problems, because protein structures are closed related to their functions. Even though chemical experiments can achieve high accuracy, they in the mean time incur high costs and long time to solve the protein problems. Hence, computational tools are then applied thereto and considered as promising ways which not only reduce the time and the costs but also maintain reliable predictive results. This motivation is triggered by the basic hypothesis that the three-dimensional (tertiary) structure of a protein is uniquely determined by its sequence of amino acids. Therefore, predicting protein structure from amino acid sequences becomes one of the most important issues in molecular biology.

The protein secondary prediction (PSS) is an intermediate and useful step for the three-dimensional (tertiary) structure prediction. To better predict secondary structure, many computational techniques have been proposed in the literature to solve the PSS prediction problem, for example: PHD, a novel prediction method proposed by Rost and Sander which uses evolutionary information and has obtained significant improvements [1]–[3]; SVM, a new method introduced by Hua and Sun which is based on statistical learning theory (SLT) [4]; and QuickRBF, a fast and innovative method proposed by Ou *et al.* [5] which is capable of delivering the same level of prediction accuracy as the LIBSVM proposed by Lin *et al.* [6], while having

execution efficiency during the phase to construct the classifier.

Despite the existence of many approaches, this issue still remains to be further studied. We propose that the single-stage approaches are unable to find complex relations (correlations) among different elements in the sequence. The result of incorporating second stage could be improved by incorporating the interactions or contextual information among the output elements of the secondary structures prediction, which are considerably reduced in their complexity. We believe it is feasible to enhance present single-stage approaches by cascading and fusing with another prediction scheme at their outputs and propose to use RBFN as the second-stage.

This thesis investigates the use of Radial Basis Function Networks for PSS prediction. We establish the QuickRBF technique based on multi-classifier to PSS prediction. Moreover, we cascade two multi-class QuickRBFs for the prediction scheme to improve the prediction accuracy from the output of the first stage. Finally, different fusion methods are applied and a high level of accuracy was achieved. We report a prediction accuracy of 76.7% on RS126 dataset based on PSI-BLAST profiles.

1.2 Introduction to the Protein Secondary Structure

Protein secondary structure prediction is to predict protein secondary structure based only on its sequence, where each amino acid is assigned a structure state, helix (**H**), strand (**E**) or coil (**C**). The secondary structure we used is assigned from the experimentally determined tertiary structure by the benchmark secondary structure definition, DSSP. According to DSSP, 8 types of protein secondary structure elements

were classified and denoted by letters: H (α-helix), E (extended β-strand), G (3_{10}-helix), I (π-helix), B (isolated β-strand), T (turn), S (bend) and "_" (rest). The 8 classes are usually reduced to three states of helix (**H**), sheet (**E**) and coil (**C**) by using one of the following methods:

1. H,G and I to **H**; E to **E**; all other states to **C**
2. H,G to **H**; E,B to **E**; all other states to **C**
3. H,G to **H**; E to **E**; all other states to **C**
4. H to **H**; E,B to **E**; all other states to **C**
5. H to **H**; E to **E**; all other states to **C**

The 8- to 3-state reduction method can alter the apparent prediction accuracy [6]. Although we can expect an accuracy increase by using method 5, we used the first method which is adopted by HYPROSP [8], [9].

The traditional three classes: α-helix, β-sheet and loop (coil) representing all the rest. The α-helix (Fig. 1.1) is the classic element of protein structure which is predicted to be stable and energetically favorable in proteins. Alpha helices in proteins are found when a stretch of consecutive residues all have the phi, psi angle pair approximately -85° and -50°, corresponding to the allowed region in the bottom left quadrant of the Ramachandran plot (Fig. 1.2).

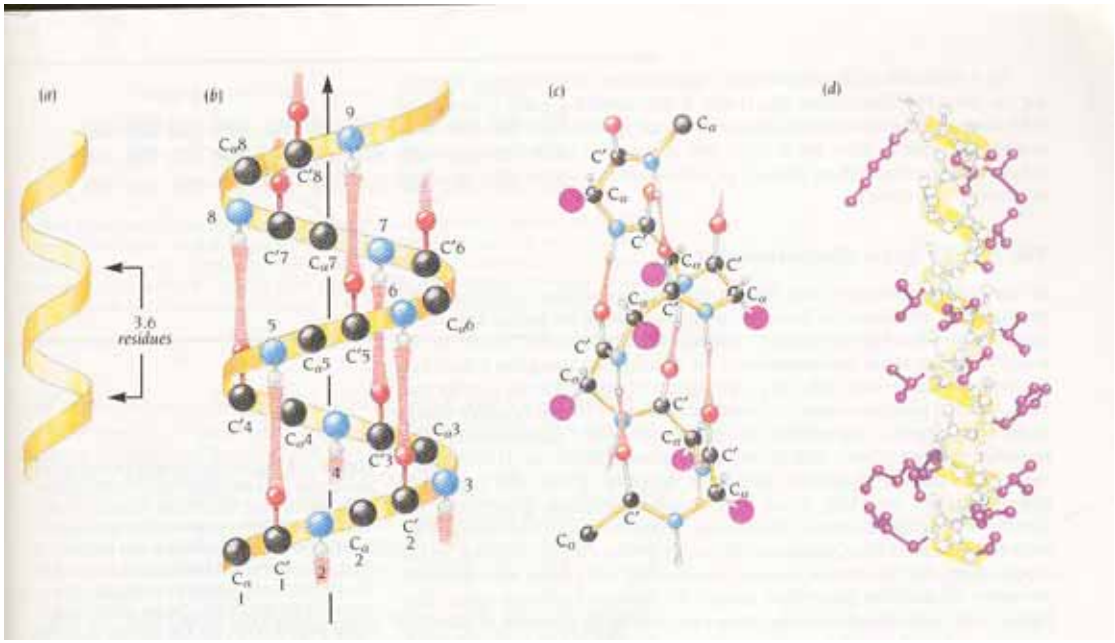


Fig. 1.1. The α-helix structure.

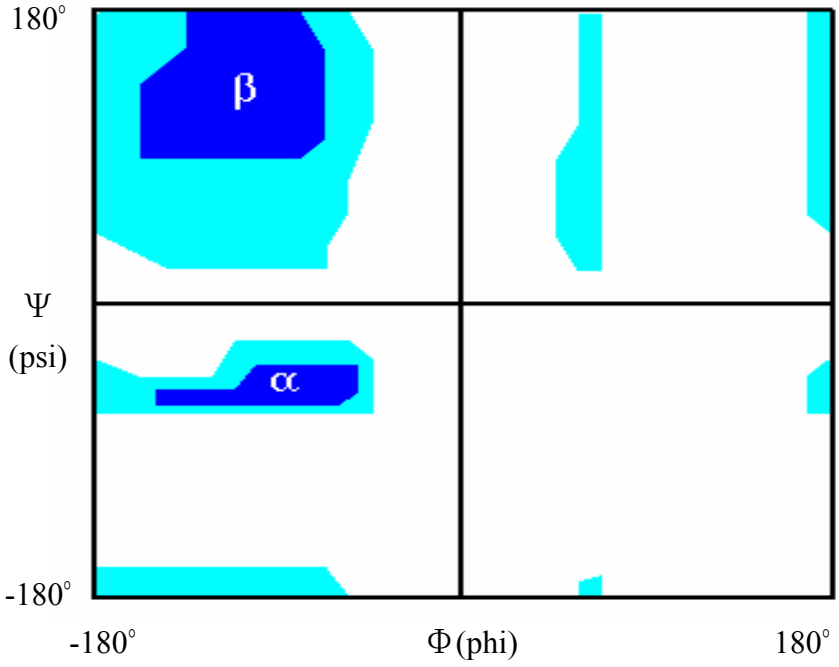


Fig. 1.2. The Ramachandran plot.

Only in the α-helix are the backbone atoms properly packed to provide a stable structure. In globular proteins, the average length for α-helices is around ten residues, corresponding to three turns. The rise per residue of an α-helix is 1.5 Å along the helical axis, which corresponds to about 15 Å from one end to the other of an average α-helix.

The second major structural element found in globular proteins is the β-sheet. This structure is built up from a combination of several regions of the polypeptide chain, in contrast to the α-helix, which is built up from one continuous region. These regions, β-strands, are usually from five to ten residues long and are in an almost fully extended conformation with phi, psi angles within the broad structurally allowed region in the upper left quadrant of the Ramachandran plot (Fig. 1.2). The β-strands can interact in two ways to form a pleated sheet – parallel and anti-parallel. Each of the two forms has a distinctive pattern of hydrogen-bonding. The anti-parallel β-sheet (Fig. 1.3) has narrowly spaced hydrogen bond pairs that alternate with widely spaced pairs. Parallel β-sheets (Fig. 1.4) have evenly spaced hydrogen bonds that bridge the β-strands at an angle.

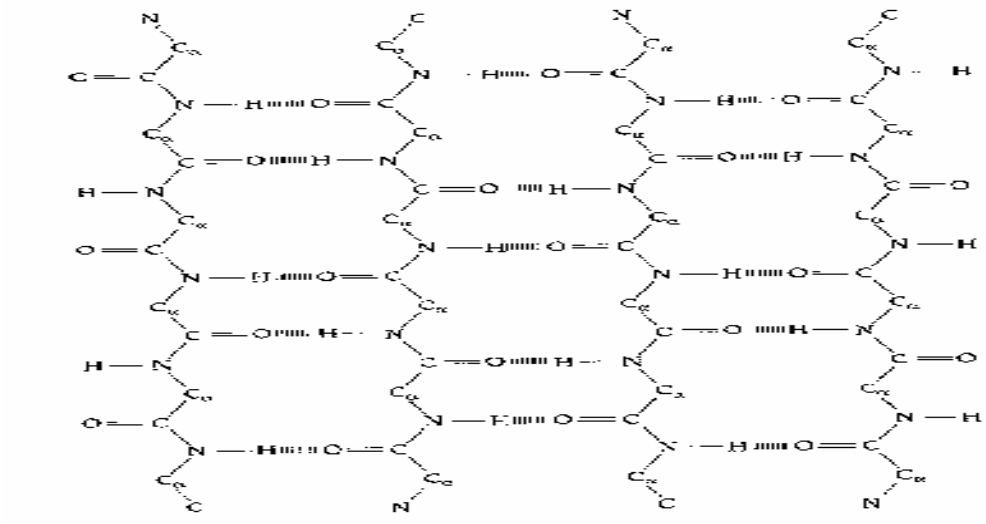


Fig. 1.3. The Anti-parallel β-sheet.

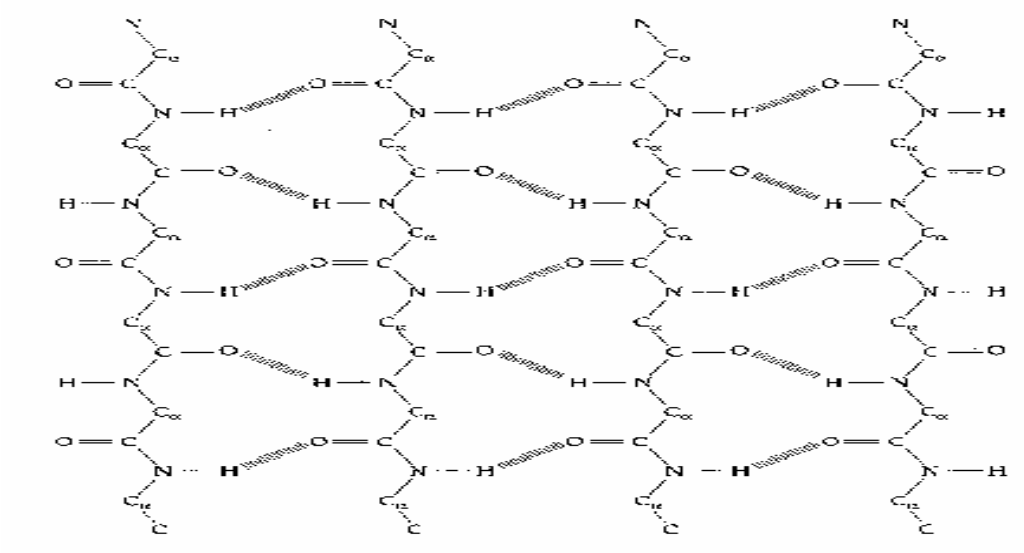


Fig. 1.4. The Parallel β-sheet.

Most protein structures are built up from combinations of secondary structure elements, α-helices and β-strands, which are connected by loop regions of various lengths and irregular shape. The loop regions are always at the surface of protein molecules. Loop regions exposed to solvent are rich in charge and polar hydrophilic residues. Loop regions that connect two adjacent anti-parallel β-strands are called the *hairpin loops*. Short hairpin loops are usually called *reverse turns* or simply *turns*.

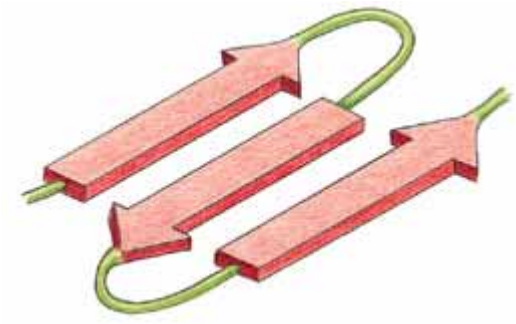


Fig. 1.5. Two hairpin loops between three anti-parallel β-strands.

1.3 Introduction to Neural Networks and Approximation Schemes

The problem of learning a mapping between an input and an output space is essentially equivalent to the problem of synthesizing an associative memory that retrieves the appropriate output when presented with the input and generalizes when presented with new inputs. It is also equivalent to the problem of estimating the system that transforms inputs into outputs given a set of examples of input-output pairs. A classical framework for this problem is Approximation Theory. Almost all approximation schemes can be mapped into some kind of network that can be dubbed as a "neural network." Networks, after all, can be regarded as a graphic notation for a large class of algorithms. In the context of this thesis, a network is a function represented by the composition of many basic functions.

To measure the quality of the approximation, one introduces a distance function ρ to determine the distance $\rho[f(\mathbf{X}), F(\mathbf{W}, \mathbf{X})]$ of an approximation $F(\mathbf{W}, \mathbf{X})$ from $f(\mathbf{X})$. The distance is usually induced by a norm, for instance the standard L_2 norm. The approximation problem can be then stated formally as:

Approximation problem: If $f(\mathbf{X})$ is a continuous function defined on set $\mathbf{X}$, and $F(\mathbf{W}, \mathbf{X})$ is an approximating function that depends continuously on $\mathbf{W} \in \mathbf{P}$ and $\mathbf{X}$, the approximation problem is to determine the parameters $\mathbf{W}^*$ such that

$$\rho\,[F(\mathbf{W}^*, \mathbf{X}),\ f(\mathbf{X})] < \rho\,[F(\mathbf{W}, \mathbf{X}),\ f(\mathbf{X})] \qquad (1.1)$$

for all $\mathbf{W}$ in the set $\mathbf{P}$.

With these definitions we can consider a few examples of $F(\mathbf{W}, \mathbf{X})$, shown in the Fig. 1.6 where (a) indicates a linear approximating function, (b) indicates polynomial estimators and other linear combinations of nonlinear features on the input, and (c) indicates a back-propagation network.

1) The classical linear case is

$$F(\mathbf{W},\ \mathbf{X}) \ = \ \mathbf{WX} \tag{1.2}$$

where $\mathbf{W}$ is an $m \times n$ matrix and $\mathbf{X}$ is an n-dimensional vector. It corresponds to a network without hidden units;

2) The classical approximation scheme is linear in a suitable basis of functions $\Phi_i(\mathbf{X})$ of the original inputs $\mathbf{X}$, that is

$$F(\mathbf{W},\ \mathbf{X}) \ = \ \mathbf{W}\,\Phi_i(\mathbf{X}) \tag{1.3}$$

and corresponds to a network with one layer of hidden units. Spline interpolation and many approximation schemes, such as expansions in series of orthogonal polynomials, are included in this representation. When the Φ_i are products and powers of the input components $\mathbf{X}_i$, F is a polynomial.

3) The nested sigmoids scheme (usually called back-propagation, BP in short) can be written as

$$F(\mathbf{W},\mathbf{X}) = \rho(\sum_{\mathrm{n}} \mathbf{w}_{\mathrm{n}}\rho(\sum_{i} \mathbf{v}_i\rho(...\rho(\sum_{j} \mathbf{u}_j\mathbf{X}_j)...))) \tag{1.4}$$

and corresponds to a multilayer network of units that sum their inputs with "weights" $\mathbf{w}$, $\mathbf{v}$, $\mathbf{u}$,… and then perform a sigmoidal transformation of this sum. This scheme (of nested nonlinear functions) is unusual in the classical theory of the approximation of continuous functions.

In general, each approximation scheme has some specific algorithm for finding the optimal set of parameters $\mathbf{W}$. An approach that works in general, though it may not be the most efficient in any specific case, is some relaxation method, such as gradient descent or conjugate gradient or simulated annealing, in parameter space, attempting to minimize the error ρ over the set of examples. In any case, our discussion suggests that networks of the type used recently for simple learning tasks can be considered as specific methods of function approximation. In this thesis, the

applied networks of the type is an efficient construction of Radial Basis Function Networks used for fast modeling tasks and can be considered as specific methods of function approximation.

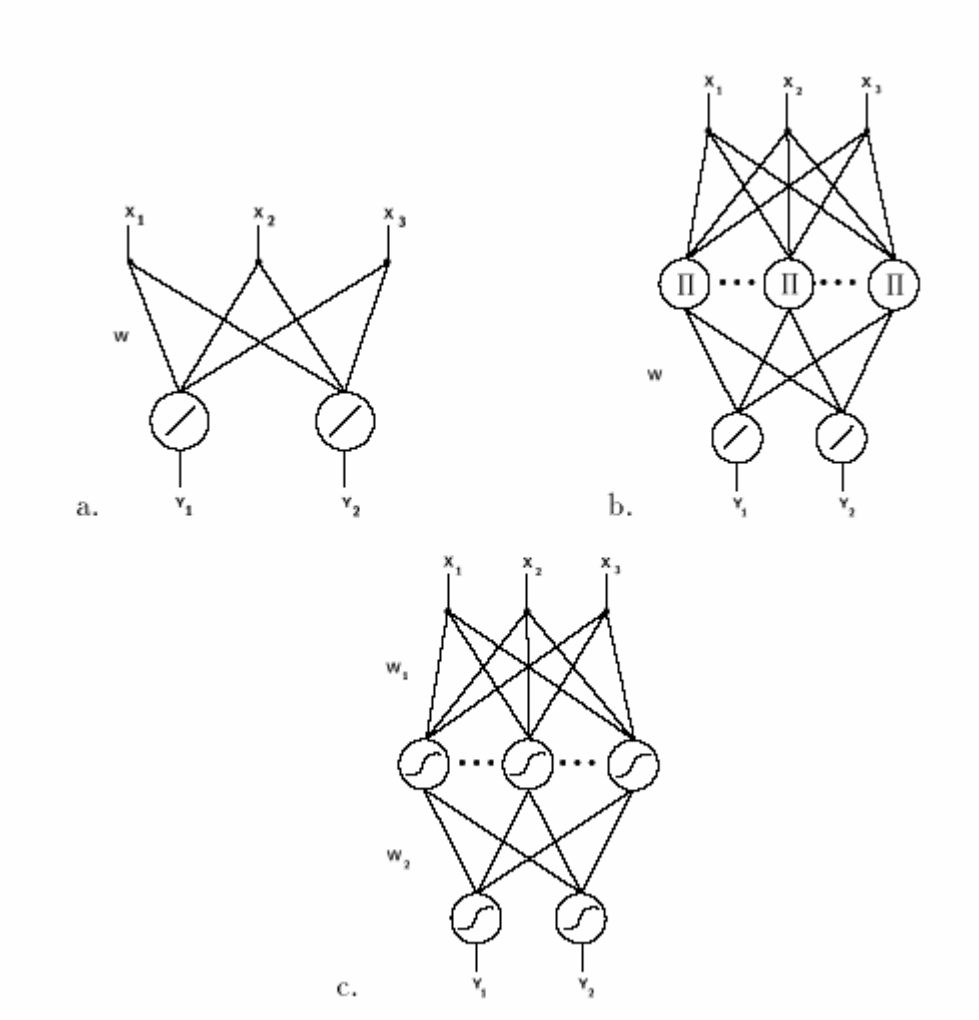
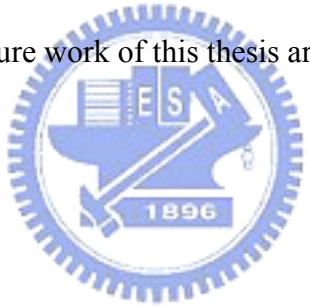


Fig. 1.6. Approximation functions with different estimators.

1.4 Thesis Outline

The organization of this thesis is structured as follows. Chapter 1 introduces the role of neural networks and the motivation and the background of this thesis. In Chapter 2, the quick radial basis function networks will be described. Moreover, we will present the fuzzy ARTMAP which was applied to a cascade of fuzzy ARTMAP and QuickRBF architecture detailed in the next chapter. We then will introduce the benchmark SVM approaches, and our paralleling QuickRBF architectures using different fusion methods in Chapter 3. In Chapter 4, the experiment of computer simulation and the results are conducted and compared to other prevailing methods, such as single stage SVM approach, dual-SVM approach and the famous PHD. Finally, the conclusion and future work of this thesis are presented in Chapter 5.



Chapter 2. Quick Radial Basis Functions

2.1 Radial Basis Functions and QuickRBF

Networks based on radial basis functions have been developed to address some of the problems encountered with training multilayer perceptrons: radial basis functions are usually able to converge and the training is much more rapid. Both are feed-forward networks with similar-looking diagrams and their applications are similar; however, the principles of action of radial basis function networks and the way they are trained are quite different from multilayer perceptrons.

An RBFN (radial basis function network) consists of three layers, namely the input layer, the hidden layer and the output layer. The input layer broadcasts the coordinates of the input vector to each of the nodes in the hidden layer. Each node in the hidden layer then produces an activation based on the associated radial basis function. Finally, each node in the output layer computes a linear combination of the activations of the hidden nodes.

For radial basis function networks, each hidden unit represents the center of a cluster in the data space. Input to a hidden unit in a radial basis function is not the weighted sum of its inputs but a distance measure: a measure of how far the input vector is from the center of the basis function for that hidden unit. Various distance measures are used, but perhaps the most common is the well-known Eculidean distance measure.

If $\mathbf{x}$ and $\boldsymbol{\mu}$ are vectors, the Eculidean distance between them is given by

$$D = \|\mathbf{x} - \boldsymbol{\mu}\| = \sqrt{\sum_i (x_i - \mu_i)^2} \tag{2.1}$$

where $\mathbf{x}$ is an input vector and $\boldsymbol{\mu}$ is the location vector of the basis function for hidden

node *j*. The hidden node then computes its outputs as a function of the distance between the input vector and its center. For the Gaussian radial basis function the hidden unit output is

$$h_j(D_j^2) = e^{-D_j^2/2\sigma_j^2} \tag{2.2}$$

where D_j is the Euclidean distance between an input vector and the location vector for hidden unit *j*; h_j is the output of hidden *j* and σ_j is a measure of the size of the cluster *j* (in statistical terms it is called the variance or the square of the standard deviation).

How an RBFN reacts to a given input stimulus is completely determined by the activation functions associated with the hidden nodes and the weights associated with the links between the hidden layer and the output layer. The general mathematical form of the output nodes in an RBFN is as follows:

$$c_j(\mathbf{x}) = \sum_{i=1}^{k} w_{ji}\, \phi\left(\|x_i - \mu_i\|; \sigma_i \right) \tag{2.3}$$

where $c_j(\mathbf{x})$ is the function corresponding to the *j*-th output unit (class *j*) and is a linear combination of *k* radial basis function $\phi(\cdot)$ with center μ_i and bandwidth σ_i. Also, $\mathbf{w}_j$ is the weight vector of class *j* and w_{ji} is the weight corresponding to the *j*-th class and *i*-th center. The general architecture of RBFN is shown as follows.

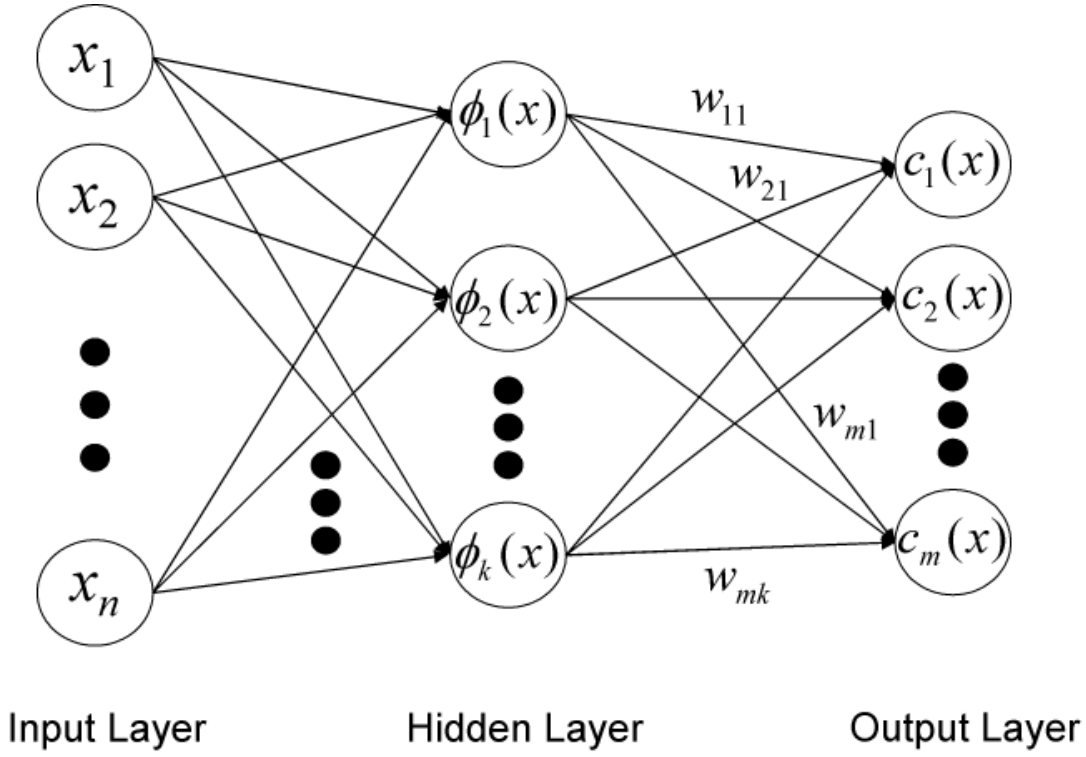


Fig. 2.1. General Architecture of Radial Basis Function Networks.

We can see that constructing an RBFN involves determining the values of three sets of parameters: the centers (μ_i), the bandwidths (σ_i) and the weights (w_{ji}), in order to minimize a suitable cost function.

In QuickRBF package, the centers are randomly selected and bandwidth are fixed and set as 5 for each kernel function for conducting the simplest method. The transformation between the inputs and the corresponding outputs of the hidden units is now fixed. The network can thus be viewed as an equivalent single-layer network with linear output units. Then, the LMSE method is used to determine the weights associated with the links between the hidden layer and the output layer.

Assume **h** is the output of the hidden layer.

$$\mathbf{h} = [\ \phi_1(\mathbf{x}),\ \phi_2(\mathbf{x}), \ldots,\ \phi_k(\mathbf{x})\ ]^{\mathrm{T}} \tag{2.4}$$

where k is the number of centers, $\phi_1(\mathbf{x})$ is the output value of first kernel function

with input **x**. Then, the discriminant function $c_j(\mathbf{x})$ of class *j* can be expressed by the following:

$$c_j(\mathbf{x}) = \mathbf{w}_j^{\mathrm{T}}\mathbf{h}, \quad j = 1, 2, \ldots, m \tag{2.5}$$

where *m* is the number of class, and $\mathbf{w}_j$ is the weight vector of class *j*. We can show $\mathbf{w}_j$ as:

$$\mathbf{w}_j = \left[\ w_{j1}(\mathbf{x}),\ w_{j2}(\mathbf{x}), \ldots,\ w_{jk}(\mathbf{x})\ \right]^{\mathrm{T}} \tag{2.6}$$

After calculating the discriminant function value of each class, we choose the class with the biggest discriminant function value as the classification result. We will discuss how to get the weight vectors by using least mean square error method in the following.

For a classification problem with *m* classes, let $\mathbf{V}_i$ designate the *i*-th column vector of an $m \times m$ identity matrix and **W** be an $k \times m$ matrix of weights:

$$\mathbf{W} = \left[\ \mathbf{w}_1,\ \mathbf{w}_2, \ldots,\ \mathbf{w}_m\ \right] \tag{2.7}$$

Then the objective function to be minimized

$$J(\mathbf{W}) = \sum_{j=1}^{m} P_j E_j \left\{ \left\| \mathbf{W}^T\mathbf{h} - \mathbf{V}_j \right\|^2 \right\} \tag{2.8}$$

where P_j and $E_j\{\cdot\}$ are the a priori probability and the expected value of class *j*, respectively.

To find the optimal **W** that minimizes *J*, the gradient of $J(\mathbf{W})$ is set to be zero:

$$\nabla_w J(\mathbf{w}) = 2\sum_{j=1}^{m} P_j E_j \left\{ \mathbf{h}\mathbf{h}^{\mathrm{T}} \right\} \mathbf{W} - 2\sum_{j=1}^{m} P_j E_j \left\{ \mathbf{h} \right\} \mathbf{V}_{\mathrm{j}}^{\mathrm{T}} = [\mathbf{0}] \tag{2.9}$$

where [**0**] is a $k \times m$ null matrix. Let $\mathbf{K}_i$ denote the class-conditional matrix of the second-order moments of **h**, i.e.

$$K_i = E_i\left\{\mathbf{h}\mathbf{h}^{\mathrm{T}}\right\} \tag{2.10}$$

If **K** denotes the matrix of the second-order moments under the mixture distribution, we have

$$\mathbf{K} = \sum_{j=1}^{m} P_j \mathbf{K}_j \tag{2.11}$$

Then Eq. (2.9) becomes

$$\mathbf{K}\mathbf{W} = \mathbf{M} \tag{2.12}$$

where

$$\mathbf{M} = \sum_{j=1}^{m} P_j E_j\left\{\mathbf{h}\right\}\mathbf{V}_j^{\mathrm{T}} \tag{2.13}$$

If **K** is nonsingular, the optimal **W** can be calculated by

$$\mathbf{W}^* = \mathbf{K}^{-1}\mathbf{M} \tag{2.14}$$

However, there is a critical drawback of this method. That is, **K** may be singular and this will crash the whole procedure. By observing the matrix $\mathbf{h}\mathbf{h}^{\mathrm{T}}$, we are aware of that the matrix $\mathbf{h}\mathbf{h}^{\mathrm{T}}$ is symmetric positive semi-definite (PSD) matrix with rank equal to 1. Since **K** is the summation of $\mathbf{h}\mathbf{h}^{\mathrm{T}}$ for each training instance, **K** is also a PSD matrix with rank smaller than *n*. However, PSD matrix may be a singular matrix, so we should add the regularization term to make sure the matrix will be invertible. In the regularization theory, it consists in replacing the objective function as follows:

$$J(\mathbf{W}) = \sum_{j=1}^{m} P_j E_j\left\{\left\|\mathbf{W}^{\mathrm{T}}\mathbf{h} - \mathbf{V}_j\right\|^2\right\} + \lambda\sum_{j=1}^{m} \mathbf{w}_j^{\mathrm{T}}\mathbf{w}_j \tag{2.15}$$

where λ is the regularization parameter.

Then the Eq. (2.12) becomes

$$\left(\mathbf{K} + \lambda\mathbf{I}\right)\mathbf{W} = \mathbf{M} \tag{2.16}$$

If we set $\lambda > 0$, $\left(\mathbf{K} + \lambda\mathbf{I}\right)$ will be a positive definite (PD) matrix and therefore is nonsingular. The optimal $\mathbf{W}^*$ can be calculated by

$$\mathbf{W}^* = \left(\mathbf{K} + \lambda\mathbf{I}\right)^{-1}\mathbf{M} \tag{2.17}$$

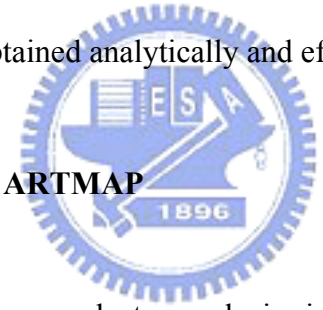
However, the PD matrix has many good properties, and one of them is a special and efficient triangular decomposition, Cholesky decomposition. By using Cholesky decomposition, we can decompose the $\left(\mathbf{K} + \lambda\mathbf{I}\right)$ matrix as follows:

$$\left(\mathbf{K} + \lambda\mathbf{I}\right) = \mathbf{L}\mathbf{L}^{\mathrm{T}} \tag{2.18}$$

where **L** is a lower triangular matrix. Then, the Eq. (2.16) becomes

$$\left(\mathbf{L}\mathbf{L}^{\mathrm{T}}\right)\mathbf{W} = \mathbf{M} \tag{2.19}$$

Actually, the linear system can be solved efficiently by using back-substitution twice. Finally, we can get the optimal $\mathbf{W}_j^*$ for class *j* from $\mathbf{W}^*$, and then the optimal discriminant function $c_j(\mathbf{x})$ for class *j* is derived. By using the regularization theory, the optimal weights can be obtained analytically and efficiently.



2.2 Fuzzy ART and Fuzzy ARTMAP

The aim of classification, or cluster analysis, is to organize observations into similar groups. Cluster analysis is a commonly used, appealing and conceptually intuitive statistical method. Some of its uses include pattern recognition, where pixels of obtained images are grouped into clusters with similar attributes for targeted objects; gene expression analysis, where genes with similar expression patterns are grouped together and so on. A cluster analysis results in a simplification of a data set for two reasons: first, because each cluster, which is now relatively homogeneous, can be analyzed separately, and second, because the data set can be summarized by a description of each cluster. Thus, it can be used to effectively reduce the size of massive amounts of data. In this thesis, a famous tool of cluster discovery, fuzzy ARTMAP, is adopted to retrieve certain features existing in the dataset, and detailed

as follows.

2.2.1 Fuzzy ART

The Fuzzy ART architecture is capable of performing unsupervised learning against either binary or analog input vectors. Basically, Fuzzy ART consists of three neural layers: preprocessing F_0, matching F_1 and competitive F_2.

Every input vector component, $\mathbf{a}_i$, must be normalized between 0 and 1. Layer F_0 is formed by $2M$ neurons, with M being the dimension of the input vectors, and provides the complement code of the input vectors according to the following expression:

$$\mathbf{I}_i = \begin{cases} \mathbf{a}_i & 1 \le i \le M \\ 1 - \mathbf{a}_{i-M} & M+1 \le i \le 2M \end{cases} \tag{2.20}$$

Layer F_1 is also formed by $2M$ neurons and its function is to verify the match between input patterns and prototypes learned by the network. Finally, layer F_2 is a competitive layer. It works as a content addressable memory (Carpenter *et al.*, 1998) where each neuron stores a prototype of a class of input vectors. F_2 is formed by a total number of N neurons which are recruited dynamically as they are needed to encode new classes of incoming vectors. Each layer is connected to the next through a set of adaptive weighted paths. These weights, W_{ij}, form the long term memory (LTM) element of the neural network and evolve during the training phase. Every weight is initialized to 1 at the beginning of the training and monotonically decreases as the training proceeds and patterns are learned. This monotonical decrease of weights guarantees the eventual stability of the network.

Unsupervised learning in Fuzzy ART is performed in the following way. Each input pattern, $\mathbf{a}$, is put into its complement code, **I**, according to Eq. (2.20), and then

it is transmitted through F_1 to layer F_2. Each neuron j in F_2 receives an activation, $T_j(\mathbf{I})$, that is a function of the input pattern and the LTM weights:

$$T_j = \frac{\left|\mathbf{I} \wedge \mathbf{W}_j\right|}{\alpha + \left|\mathbf{W}_j\right|} \qquad j = 1,\ldots,N \tag{2.21}$$

where $\mathbf{W}_j = \left\lfloor W_{j1}, W_{j2}, \ldots, W_{j2M} \right\rfloor$ are the weights associated with neuron j; $\left|\cdot\right|$ is the L^1 norm, $\left|\mathbf{x}\right| = \sum_{i=1}^{M} x_i$; $x \wedge y = \min\{x, y\}$ is the fuzzy AND operator, for vectors, $\mathbf{x} \wedge \mathbf{y} = \mathbf{v}$ with $\mathbf{v} = \min\{\mathbf{x}, \mathbf{y}\}$ and $\alpha \in [0, \infty]$ is a choice parameter (typically $\alpha \approx 0^+$).

At this point, the neurons in F_2 hold a WTA competition to select which neuron, *J*; is going to learn the pattern:

$$J = \arg\max_j \{T_j\} \tag{2.22}$$

After the competition, only the output of the winning neuron remains set to 1 and descends through the top-down weighted paths so that the prototype of neuron *J* is presented in layer F_1. In F_1 the matching between the input pattern, **I**; and the winner prototype, $\mathbf{W}_j$, is evaluated according to a criterion determined by a user defined parameter $\rho \in [0, 1]$. The criterion is applied as follows:

1) If $\frac{\left|\mathbf{I} \wedge \mathbf{W}_J\right|}{\left|\mathbf{I}\right|} \geq \rho$, then the input is considered to belong to match prototype in *J* and pattern is learn by neuron *J*.

2) If $\frac{\left|\mathbf{I} \wedge \mathbf{W}_J\right|}{\left|\mathbf{I}\right|} < \rho$, then the system is reset and neuron *J* is inhibited so that it no longer enters the competition for the current pattern. In addition to this, a match tracking mechanism raises the value of parameter ρ so that the next winner must be closer to the pattern. After this new competition,

another winner is selected. Eventually, a new neuron in F_2 will be committed if none of the current neurons is found to match the pattern sufficiently.

When a winner successfully passes the matching criterion, learning occurs. LTM weights are updated according to the following learning law:

$$\mathbf{W}_J^{\text{New}} = \beta\left(\mathbf{W}_J^{\text{Old}} \wedge \mathbf{I}\right) + \left(1-\beta\right)\mathbf{W}_J^{\text{Old}} \tag{2.23}$$

where $\beta \in [0,1]$ is the learning rate: $\beta \to 0^+$ implies slow learning, while $\beta = 1$ implies fast learning and each pattern is incorporated to the knowledge stored by the network in just one iteration.

With complement coding of patterns and the L^1 norm, each F_2 neuron can be represented geometrically as a hyperbox in R^M covering all the patterns that it has already learned. The size of the hyperbox R_j associated with neuron j, is determined by weights $\mathbf{W}_j$ as showed in Fig. 2.2. Competition in layer F_2 has also a geometric interpretation. Activation function, T_j, is a measure of the distance between the pattern $\mathbf{a}$ and R_j (Fig. 2.2). Therefore, the neuron with the box lying nearest to the pattern will receive the highest activation. Parameter α in Eq. (2.21) is used to break ties when several boxes include the pattern; in such case, the smaller the box is, the higher the activation received.

Finally, the learning process can be viewed as the expansion of the winner neuron box toward the pattern. If fast learning is applied, the box grows until it actually covers the pattern, while under slow learning the box just expands toward the pattern but without covering it.

Referring to Fig. 2.2, it shows the geometric interpretation of Fuzzy ART. Box R_j is associated with neuron j, while $\mathbf{I} = [I_1, I_2, I_3, I_4]$ is the complement code of

input pattern **a** . Size of box R_j is determined by weighs associated with neuron *j*, $\mathbf{W}_j = \left[W_{1j}, W_{2j}, W_{3j}, W_{4j} \right]$. In a generic *M* dimensions case, R_j size on dimension **I** is determined by W_{ij} and $W_{i+M,j}$.

ART 1 (Binary)	FUZZY ART (Analog)

Category Choice

$$T_j = \frac{|\mathbf{I} \cap \mathbf{W}_j|}{\alpha + |\mathbf{W}_j|} \qquad T_j = \frac{|\mathbf{I} \wedge \mathbf{W}_j|}{\alpha + |\mathbf{W}_j|}.$$

Match Criterion

$$\frac{|\mathbf{I} \cap \mathbf{W}_j|}{|\mathbf{I}|} \geq \rho \qquad \frac{|\mathbf{I} \wedge \mathbf{W}_j|}{|\mathbf{I}|} \geq \rho$$

Fast Learning

$$\mathbf{W}_j^{(\text{new})} = \mathbf{I} \cap \mathbf{W}_j^{(\text{old})} \qquad \mathbf{W}_j^{(\text{new})} = \mathbf{I} \wedge \mathbf{W}_j^{(\text{old})}$$

$\cap$ = logical AND intersection $\wedge$ = fuzzy AND minimum

Fig. 2.2. Comparison of ART1 and fuzzy ART.

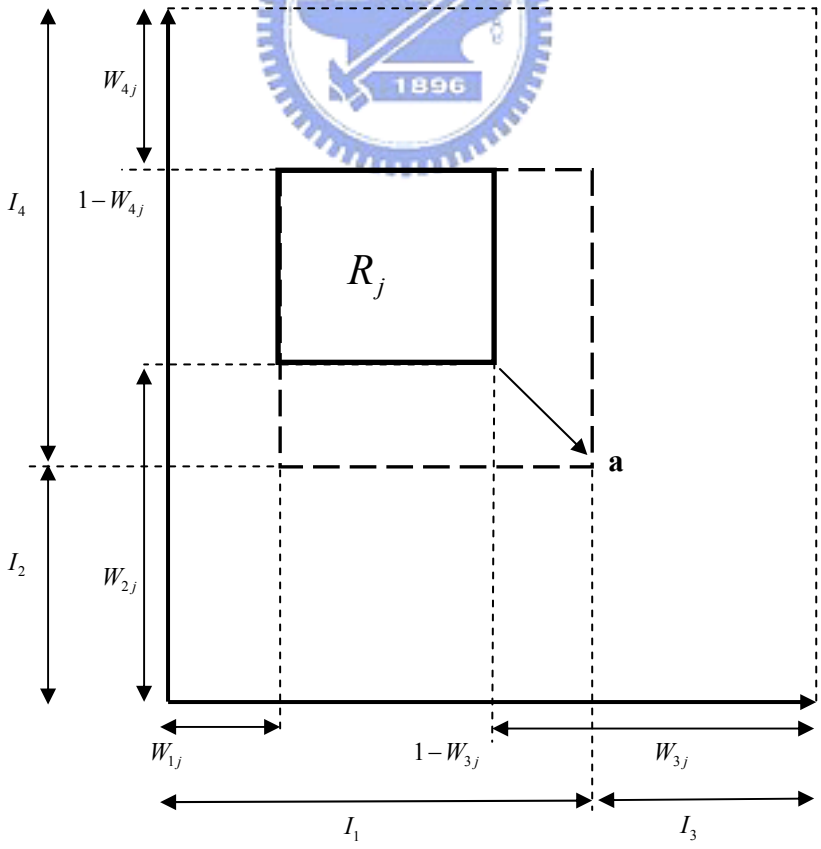


Fig. 2.3. Geometric interpretation of Fuzzy ART.

2.2.2 Fuzzy ARTMAP

Fuzzy ARTMAP is an incremental supervised learning of recognition categories and multidimensional maps in response to arbitrary sequences of analog or binary input vectors, which may represent fuzzy or crisp sets of features. It realizes a new minimax learning rule that conjointly minimizes predictive error and maximizes code compression, or generalization. It automatically learns a minimal number of recognition categories, or hidden units, which is achieved by a match tracking process.

The applications of ARTMAP involve analog patterns that are not necessarily interpreted as fuzzy set, but serve to illustrate the properties of the system and allow comparison with several existing systems, such as the benchmark back-propagation and genetic algorithm systems. In all cases, fuzzy ARTMAP simulations lead to favorable levels of learned predictive accuracy, speed, and code compression in both on-line and off-line setting [11].

The fuzzy ARTMAP system incorporates two fuzzy ART modules, $\mathrm{ART_a}$ and $\mathrm{ART_b}$, that are linked together via an inter-ART module, F^{ab}, called a *map field*. The map field is used to form predictive associations between categories and to realize the *match tracking rule*, whereby the vigilance parameter of $\mathrm{ART_a}$ increases in response to a predictive mismatch at $\mathrm{ART_b}$. Match tracking reorganizes category structure so that predictive error is not repeated on subsequent presentations of the input. The interactions mediated by the map field F^{ab} may be operationally characterized as follows.

ART_a and ART_b: Inputs to $\mathrm{ART_a}$ and $\mathrm{ART_b}$ are in the complement code form: for $\mathrm{ART_a}$, $\mathbf{I} = \mathbf{A} = \left[\mathbf{a}, \mathbf{a}^c\right]$; and for $\mathrm{ART_b}$, $\mathbf{I} = \mathbf{B} = \left[\mathbf{b}, \mathbf{b}^c\right]$ (Fig. 2.4). Variables in

$\mathrm{ART_a}$ or $\mathrm{ART_b}$ are designated by subscripts or superscripts *a* and *b*. For $\mathrm{ART_a}$, let $\mathbf{x}^a \equiv \left[x_1^a, x_2^a, \cdots, x_{2M_a}^a\right]$ denote the F_1^a output vector; let $\mathbf{y}^a \equiv \left[y_1^a, y_2^a, \cdots, y_{N_a}^a\right]$ denote the F_2^a output vector; and let $\mathbf{w}_j^a \equiv \left[w_{j1}^a, w_{j2}^a, \cdots, w_{j,2M_a}^a\right]$ denote the *j*-th $\mathrm{ART_a}$ weight vector. For $\mathrm{ART_b}$, let $\mathbf{x}^b \equiv \left[x_1^b, x_2^b, \cdots, x_{2M_b}^b\right]$ denote the F_1^b output vector; let $\mathbf{y}^b \equiv \left[y_1^b, y_2^b, \cdots, y_{N_b}^b\right]$ denote the F_2^b output vector; and let $\mathbf{w}_k^b \equiv \left[w_{k1}^b, w_{k2}^b, \cdots, w_{k,2M_b}^b\right]$ denote the *k*-th $\mathrm{ART_b}$ weight vector. For the map field, let $\mathbf{x}^{ab} \equiv \left[x_1^{ab}, x_2^{ab}, \cdots, x_{N_b}^{ab}\right]$ denote the F_{ab} output vector; and let $\mathbf{w}_j^{ab} \equiv \left[w_{j1}^{ab}, w_{j2}^{ab}, \cdots, w_{j,N_b}^{ab}\right]$ denote the weight vector from the *j*th F_2^a node to F^{ab}. Vectors $\mathbf{x}^a$, $\mathbf{y}^a$, $\mathbf{x}^b$, $\mathbf{y}^b$, and $\mathbf{x}^{ab}$ are set to $\mathbf{0}$ during input presentations.

Map Field Activation: The map field F^{ab} is activated whenever one of the $\mathrm{ART_a}$ or $\mathrm{ART_b}$ categories is active. If node *J* of F_2^a is chosen, then its weights $\mathbf{w}_j^{ab}$ activate F^{ab}. If node *K* in F_2^b is active, then the node *K* in F^{ab} is activated by 1 to 1 pathways between F_2^b and F^{ab}. If both $\mathrm{ART_a}$ and $\mathrm{ART_b}$ are active, then F^{ab} becomes active only if $\mathrm{ART_a}$ predicts the same category as $\mathrm{ART_b}$ via the weights $\mathbf{w}_j^{ab}$. The F^{ab} output vector $\mathbf{x}^{ab}$ obeys

$$\mathbf{x}^{ab} = \begin{cases} \mathbf{y}^b \wedge \mathbf{w}_J^{ab} & \text{if the } J\text{th } F_2^a \text{ node is active and } F_2^b \text{ is active} \\ \mathbf{w}_J^{ab} & \text{if the } J\text{th } F_2^a \text{ node is active and } F_2^b \text{ is inactive} \\ \mathbf{y}^b & \text{if } F_2^a \text{ is inactive and } F_2^b \text{ is active} \\ \mathbf{0} & \text{if } F_2^a \text{ is inactive and } F_2^b \text{ is inactive} \end{cases} \tag{2.24}$$

By Eq. (2.24), $\mathbf{x}^{ab} = \mathbf{0}$ if the prediction $\mathbf{w}_j^{ab}$ is disconfirmed by $\mathbf{y}^b$. Such a mismatch event triggers an $\mathrm{ART_a}$ search for a better category, as follows.

Match Tracking: At the start of each input presentation the $\mathrm{ART_a}$ vigilance parameter ρ_a equals a baseline vigilance, $\overline{\rho}_a$. The map field vigilance parameter is ρ_{ab}. If

$$\left|\mathbf{x}^{ab}\right| < \rho_{ab}\left|\mathbf{y}^{b}\right| \tag{2.25}$$

then ρ_a is increased until it is slightly larger than $\left|\mathbf{A}\wedge\mathbf{w}_J^a\right|\left|\mathbf{A}\right|^{-1}$, where $\mathbf{A}$ is the input to F_1^a, in complement coding form. Then

$$\left|\mathbf{x}^{a}\right| = \left|\mathbf{A}\wedge\mathbf{w}_J^a\right| < \rho_b\left|\mathbf{A}\right| \tag{2.26}$$

where J is the index of the active F_2^a node. When this occurs, $\mathrm{ART_a}$ search leads either to activation of another F_2^a node J with

$$\left|\mathbf{x}^{a}\right| = \left|\mathbf{A}\wedge\mathbf{w}_J^a\right| \geq \rho_b\left|\mathbf{A}\right| \tag{2.27}$$

and

$$\left|\mathbf{x}^{ab}\right| = \left|\mathbf{y}^{b}\wedge\mathbf{w}_j^{ab}\right| \geq \rho_{ab}\left|\mathbf{y}^{b}\right| \tag{2.28}$$

or, if no such node exists, to the shutdown of F_2^a for the remainder of the input presentation.

Map Field Learning: Learning rules determine how the map field weights w_{jk}^{ab} change through time. Initially all template weights are set to 1, and learning proceeds as follows:

$$\mathbf{w}_j^{(\mathrm{New})} = \beta\left(\mathbf{I}\wedge\mathbf{w}_j^{(\mathrm{Old})}\right) + (1-\beta)\left(\mathbf{I}\wedge\mathbf{w}_j^{(\mathrm{Old})}\right) \tag{2.29}$$

where β is the learning parameter.

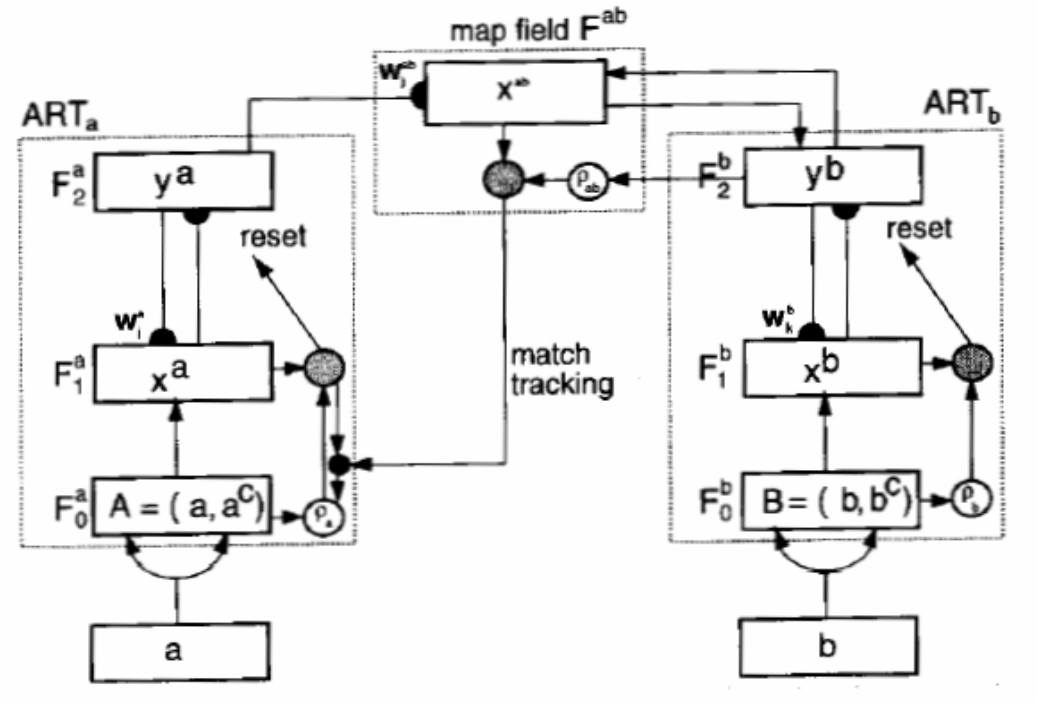


Fig. 2.4. Fuzzy ARTMAP Architecture.

During the training of radial basis function, the mean values for the K basis functions are first randomly selected and replaced by a new grouped mean after distance calculation and category assignation. In this thesis, we have chosen the geometry means, generated from the resulting categories of fuzzy ARTMAP, as the center locations to replace the randomly selected and fixed entities, chosen from the training set.

Chapter 3. Protein Secondary Structure Prediction

3.1 Support Vector Machine

The SVM is a new machine learning method that developed rapidly and has been widely used in many kinds of pattern recognition problems. The basic method of SVM is to transform the samples into a high-dimension Hilbert space and to seek a separating hyperplane in this space. The separating hyperplane, which is called the optimal separating hyperplane (OSH), is chosen in such a way as to maximize its distance from the closest training samples. As a supervised machine learning technology, SVM is well-founded theoretically on statistical learning theory. SVM has been successfully applied to many fields of pattern recognition, including object recognition, speaker identification and text categorization. The SVM usually outperforms other machine learning technologies, including Neural Networks and K-Nearest Neighbor classifiers. In recent years, the SVM has been used in bioinformatics, including gene expression profile classification, detection of remote protein homologies and recognition of translation initiation sites. Hua and Sun [4] used a single-layer SVM to analyze protein secondary structure with excellent prediction results. Sun *et al.* [12] describe a dual-layer SVM system used to predict secondary structure. The dual-layer SVM system combined with the PSI-BLAST profiles provides more accurate prediction than Hua and Sun's [4] simple SVM prediction system.

3.2 A Dual-Layer SVM Approach

A high-performance method was developed for protein secondary structure prediction based on the dual-layer support vector machine (SVM) and position-specific scoring matrices (PSSMs). SVM is a new machine learning technology that has been successfully applied in solving problems in the field of bioinformatics. The SVM's performance is usually better than that of traditional machine learning approaches. The performance was further improved by combining PSSM profiles with the SVM analysis.

However, single-stage approaches are unable to find complex relations among different elements in the sequence. So the results could be improved by incorporating the interactions or contextual information among the elements of the output sequence of secondary structures. Sun *et al.* [12] proposed a dual-layer SVM which tops the overall per-residue accuracy, Q3, at 75.2% on the CB513 data set.

As with Hua and Sun's work [4], the present analysis used the classical local coding scheme of the protein sequences with a sliding window. PSI-BLAST with n rows and 20 columns can be defined for single sequence with n residues. For the first layer in the prediction system, each residue is coded as a 21-dimensional vector, where the first 20 elements of the vector are the corresponding elements in PSI-BLAST matrix and the last unit was added to represent the N- and C-terminus. For the second layer, the vector corresponding to a residue has 4 elements, where the first 3 elements represent the 3 secondary structures (**H**, **E**, **C**). If the window length is l, the dimension of the feature vector is $21*l$ for the first layer and $4*l$ for the second layer.

A dual-layer SVM structure was used in the prediction system (see Fig. 3.1). The first layer is an SVM classifier that classifies each residue of each sequence into the 3

secondary structure classes (**H**, **E**, or **C**). The one-against-rest strategy was used for the multiclass classification, so there were three outputs for each residue. The outputs represent the probability that the residue belongs to that class. Since the consecutive patterns are correlated (e.g., a helix contains at least 4 consecutive patterns, and a sheet contains at least 3 consecutive patterns), the second-layer SVM classifier filtered successive outputs from the first layer. The target outputs of the second layer were the same as the first layer. As with the first-layer SVM, the second layer also uses the one-against-rest strategy, with each residue classified into the class with the largest output value.

This analysis used the radial basis function (RBF) kernel in both the first- and the second-layer SVM, where γ is a parameter to be determined. The analysis used the soft-margin SVM, so the regularization parameter C also needed to be regulated. γ_1 and C_1 were defined as the gamma parameter and the regularization parameter in the first-layer SVM, while γ_2 and C_2 were defined as the gamma parameter and the regularization parameter in the second-layer SVM. For the CB513 data set, $\gamma_1 = 0.05$, $C_1 = 2.3$; and $\gamma_2 = 2.5$, $C_2 = 2.0$.

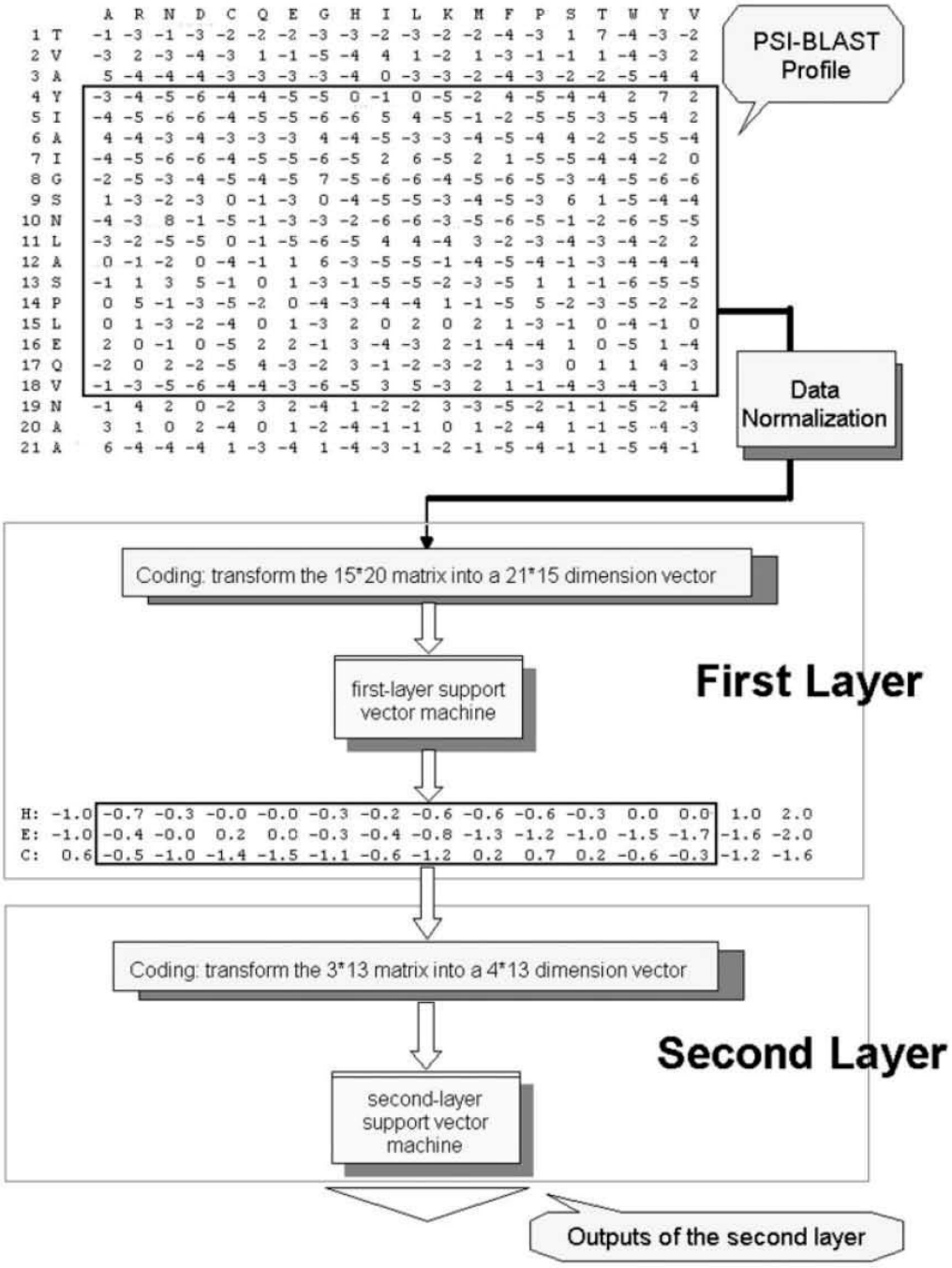


Fig. 3.1. The dual-layer SVM architecture.

3.3 Quick Radial Basis Function

Protein secondary structure prediction has been tackled by numerous learning algorithms including neural networks, SVM and other famous classifiers, and therefore presents as a classic problem for testing the effectiveness of new techniques.

The QuickRBF package, proposed by Ou *et al.* [5], can be used to conduct the experiments on the most famous data set used in protein secondary structure prediction, RS126. The RS126 data has been well studied in many publications. Also, the same 7-fold partition used by Riis and Krogh [13] is adopted.

According to the experiments done by Ou *et al.* [5], which conducted both the LIBSVM, proposed by Lin *et al.* [6], and QuickRBF approaches in the same environment and the same data sets. The detailed accuracy results [5] can be seen in Table 3.1. As the Table shows, the QuickRBF method basically delivers the same level of accuracy with LIBSVM.

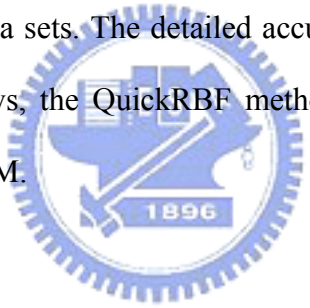


Table 3.1: Comparison of classification accuracy of the RS126 data set with PSI-BLAST PSSM profiles

RS126	LIBSVM	QuickRBF	QuickRBF	QuickRBF	QuickRBF
Centers		All	12000	5000	1000
Set A	74.06	74.14	74.01	73.73	72.71
Set B	77.44	77.01	76.32	75.54	74.76
Set C	74.99	75.01	75.07	74.93	73.85
Set D	73.11	73.69	73.72	72.44	71.44
Set E	74.08	74.19	74.26	73.97	73.14
Set F	76.93	77.23	77.39	77.28	76.12
Set G	73.82	74.27	74.30	74.07	74.36
Average	74.92	75.08	75.01	74.57	73.77

3.4 A Cascade of Fuzzy ARTMAP and QuickRBF Approach

An important performance measure of a machine learning algorithm is its generalization capability. Generalization is characterized by the number of unseen examples correctly predicted by a learning algorithm given sample training data from which to learn. One way of increasing a learning algorithm's generalization ability is to reduce its error on training data while providing it training data highly representative of the unknown target function. Fuzzy ARTMAP is designed to realize a new minimax learning rule that conjointly minimizes predictive error and maximized generalization, meaning that the system can learn to create the minimal number recognition categories or "hidden nodes" needed to meet the accuracy criterion (least prediction error).

During the training of radial basis function, the mean values for the K basis functions are first randomly selected and replaced by a new grouped mean after distance calculation and category assignation. In this thesis, we take the advantage of the fuzzy ARTMAP which is capable of learning the "hidden units" automatically. Specifically, we have chosen the geometry means of categories resulting from the fuzzy ARTMAP as the center locations instead of the randomly selected and fixed entities chosen from the training set. Therefore, a cascade of fuzzy ARTMAP and QuickRBF approach is proposed and described in this section.

Figure 3.2 shows the geometry mean of a representative category according to Eq. (3.1) when $M = 2$ in this case.

$$\left(C_{1j}, C_{2j}, \ldots, C_{Mj}\right) = \frac{1}{2}\left(W_{1j} + \left(1 - W_{M+1,j}\right),\ W_{2j} + \left(1 - W_{M+2,j}\right), \ldots,\ W_{Mj} + \left(1 - W_{2M,j}\right)\right) \quad (3.1)$$

Referring to Fig. 3.3, the inputs **a** of ART_a are representative vectors of $21*w$ dimension. The ART_a complement-coding preprocessor transforms the

21*w-dimensional vector **a** into the 2*21*w-dimensional vector $\mathbf{A} = (\mathbf{a}, \mathbf{a}^c)$ at the $\mathrm{ART_a}$ layer $\mathrm{F_0^a}$. **A** is the input vector to the $\mathrm{ART_a}$ layer $\mathrm{F_1^a}$. Similarly, the input to $\mathrm{F_1^b}$ is the 2*3-dimensional vector $\mathbf{B} = (\mathbf{b}, \mathbf{b}^c)$ where **b** represents for classes **C**, **E**, **H** encoded as (1 0 0), (0 1 0) and (0 0 1), respectively. The $\mathrm{ART_a}$ vigilance parameter ρ_a can be adjusted so that the resulting categories are accordingly changed. Similarly, other parameters such as ρ_b, ρ_{ab} and β can also alter the learning of the neural network.

A geometry mean procedure is then used to calculate the geometry means of each category. In this thesis, this geometry means are applied to stand for the centers of each category. Replacing the randomly selected centers by the geometry means produced from the fuzzy ARTMAP, the cascade architecture of fuzzy ARTMAP and QuickRBF is achieved as shown in Fig. 3.3.

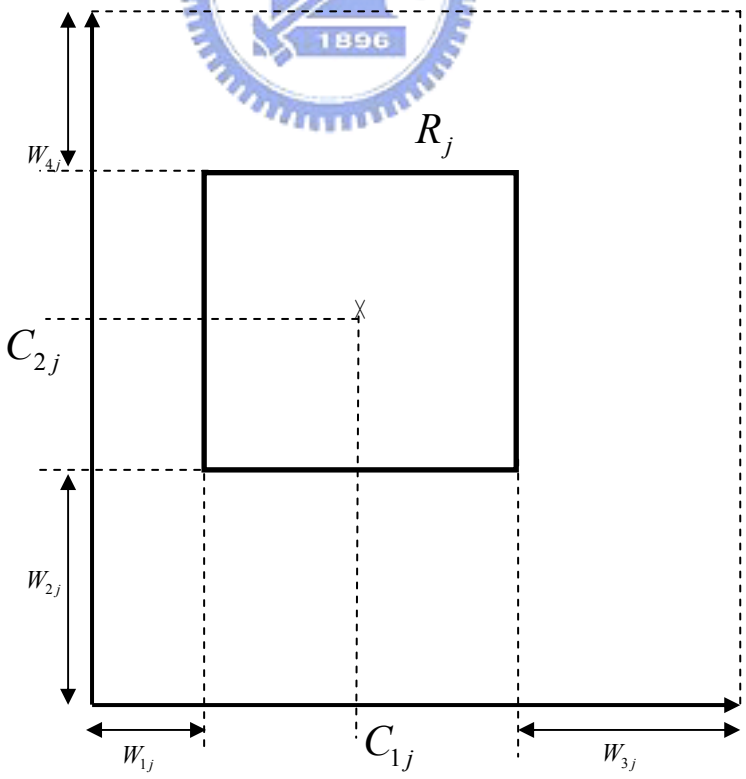


Fig. 3.2. Geometry mean of hyperbox R_j

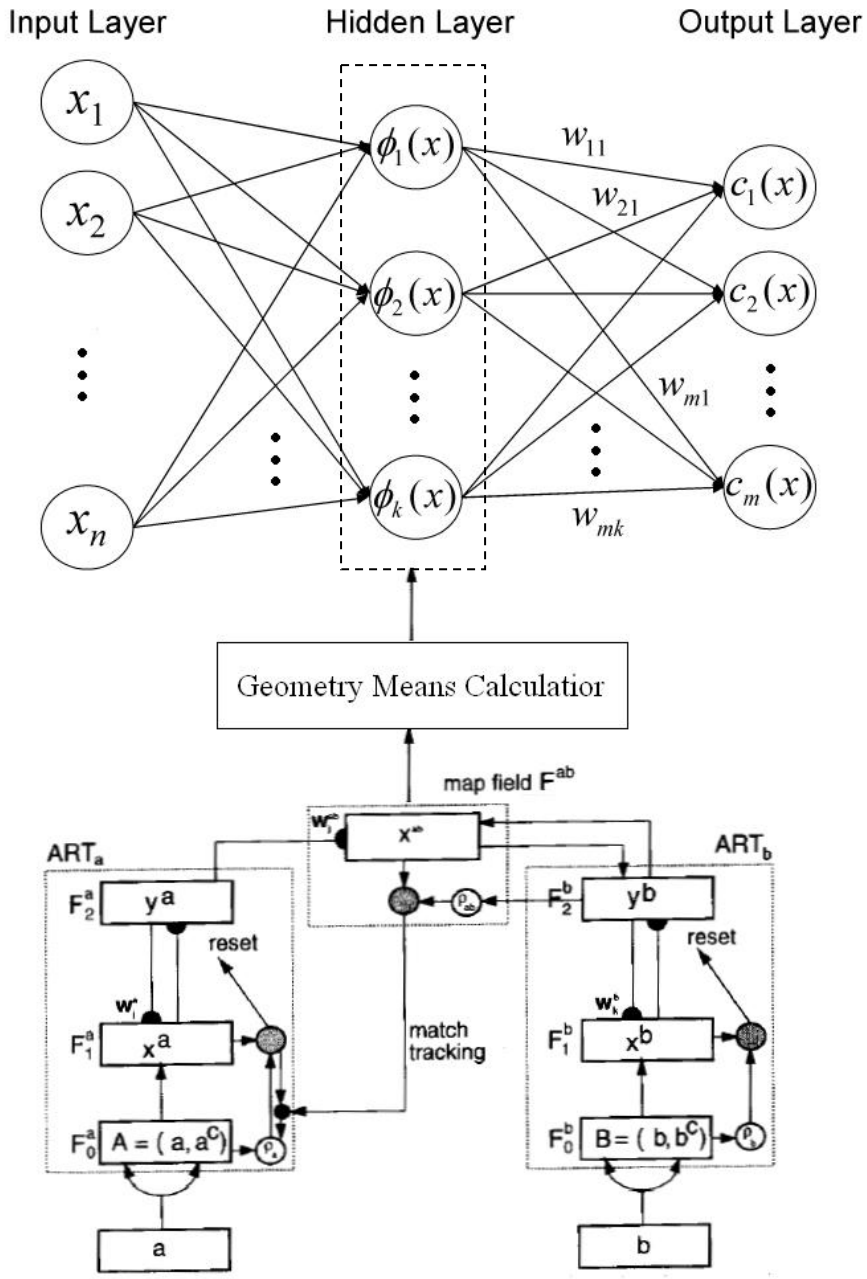


Fig. 3.3. The cascade of Fuzzy ARTMAP and QuickRBF architecture.

3.5 A Dual-Layer QuickRBF Approach

The efficient method was developed for protein secondary structure prediction based on the QuickRBF approaches. QuickRBF is an innovative neural network technology that is capable of delivering the same level of prediction accuracy as the SVM, while enjoying execution efficiency during the phase to construct the classifier.

In this thesis, a dual-layer QuickRBF is conducted as shown in Fig. 3.4. The first layer is a QuickRBF classifier which maps the 21*w dimension representative vector into the 3 classes of PSS (**H**, **E** or **C**). Instead of outputting the class labels, the values of individual output nodes which can be regarded as the probability that the residue belongs to that class for each residue are fed into the second layer after treating the same coding scheme as Hua and Sun [4]. Specifically, the second layer QuickRBF classifies the 4*l vectors and the target outputs of the second layer were the same as the first layer. Finally, each residue is classified into the class with the largest output value.

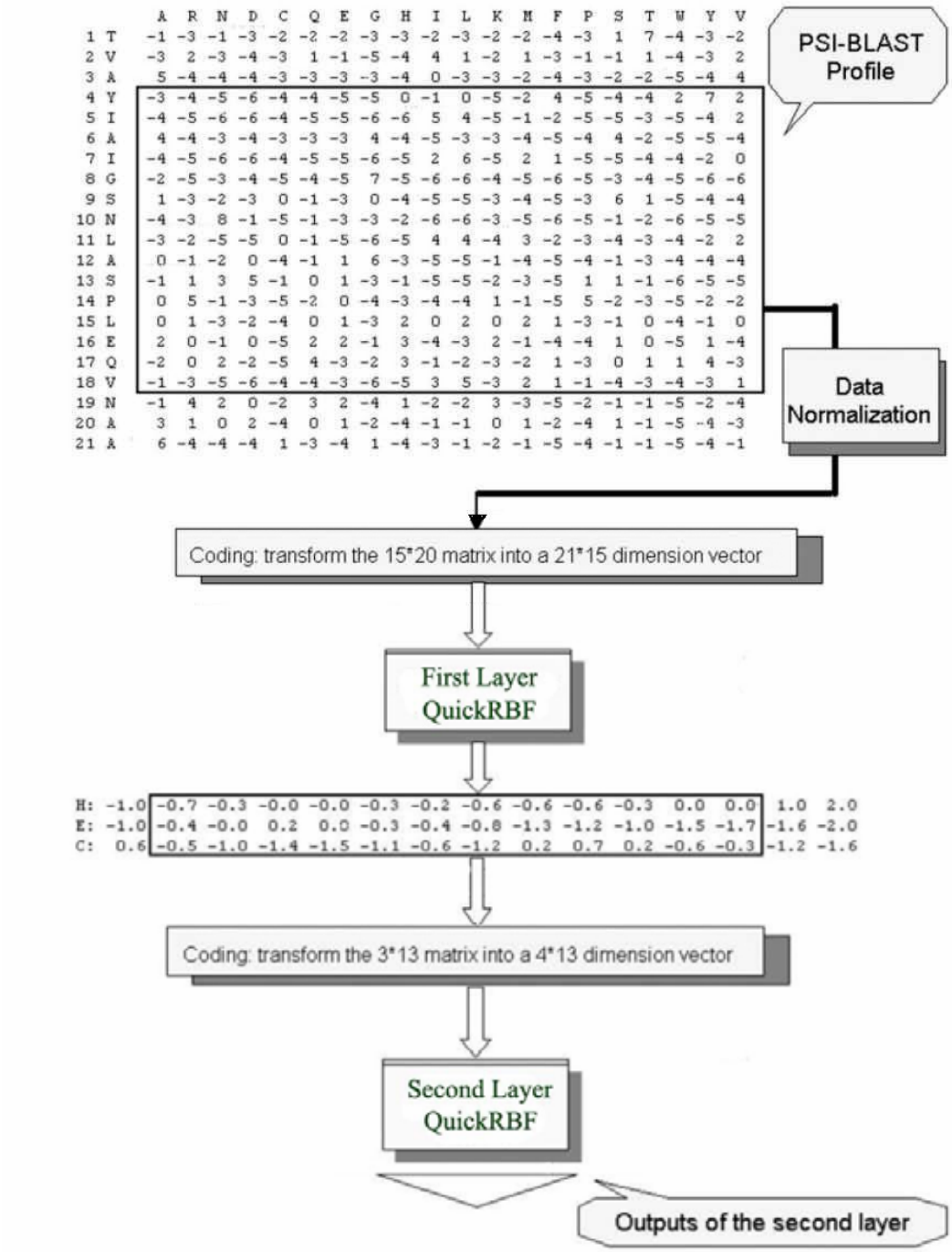


Fig. 3.4. The dual-layer QuickRBF architecture.

3.6 Fusion Method

3.6.1 Reliability Index

The prediction reliability index (RI) was used to access the effectiveness of the approaches for the prediction of the secondary structure of a new sequence. The RI offers an excellent tool for focusing on key regions having high prediction accuracy. There are different definitions of the RI. Here we used a definition similar to that proposed by Rost and Sander: RI = maximal_output(I)－Second_largest_output(I). If the value of RI > 0.9, then set RI = 0.9, so the value of RI is between 0 and 0.9. The distribution of the prediction accuracy with different RIs is illustrated in Fig. 3.5. The prediction accuracy of residues with higher RI values is much better than those with lower RI values. Therefore, the definition of RI reflects the prediction reliability.

In this research, to combine the output from the first and second layer, we developed a new classifier design using RI. In this scheme, the output with the maximum RI is chosen as the representative classifier for the final decision of the class. Based on the largest output value of this representative classifier, the final class is chosen. For example, if the output values of the decision function of each classifiers (first layer: **C**/**E**/**H**, second layer: **C**/**E**/**H**) are 0.1/0.2/0.7 and 0.3/0.2/0.5, their RI s are 0.5 (0.7－0.2) and 0.2 (0.5－0.3) respectively. Therefore, the output with highest RI, here the first layer, can be chosen for deciding the final class. Once this representative classifier is selected, the final class is assigned based on the output value of this classifier. In this example, since the largest value of the first layer is the third node, the final class if assigned as helix.

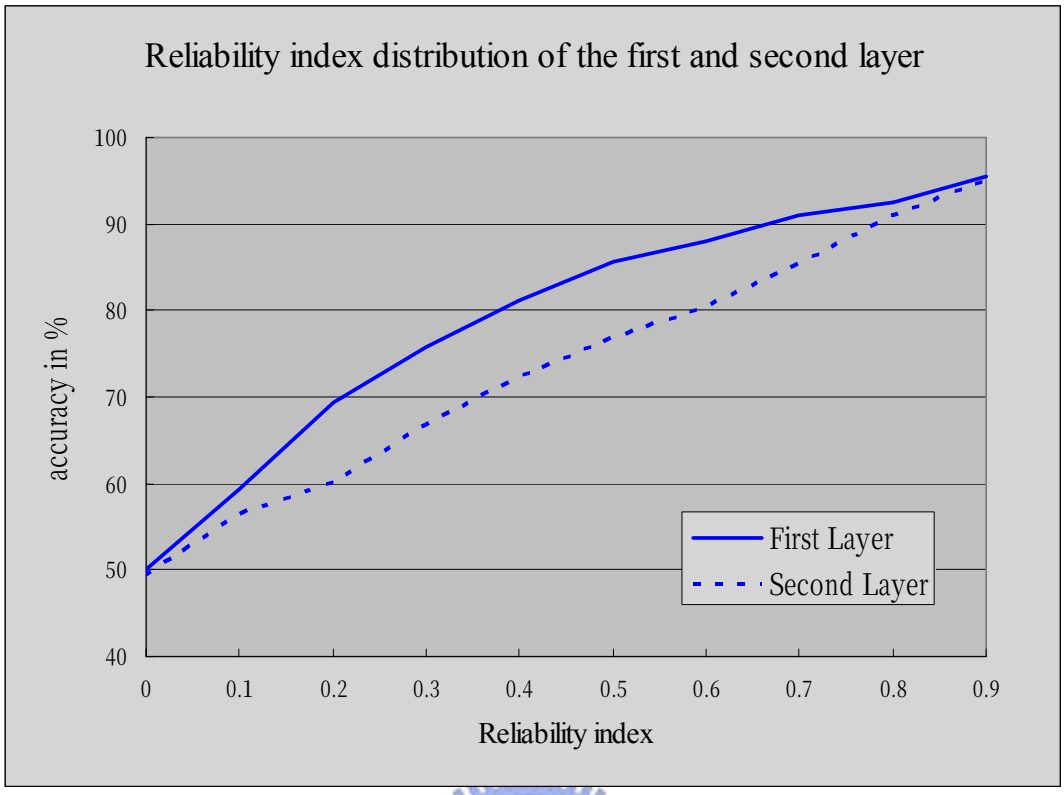


Fig. 3.5. The accuracy distribution on different Reliability indices (from 0 to 0.9).

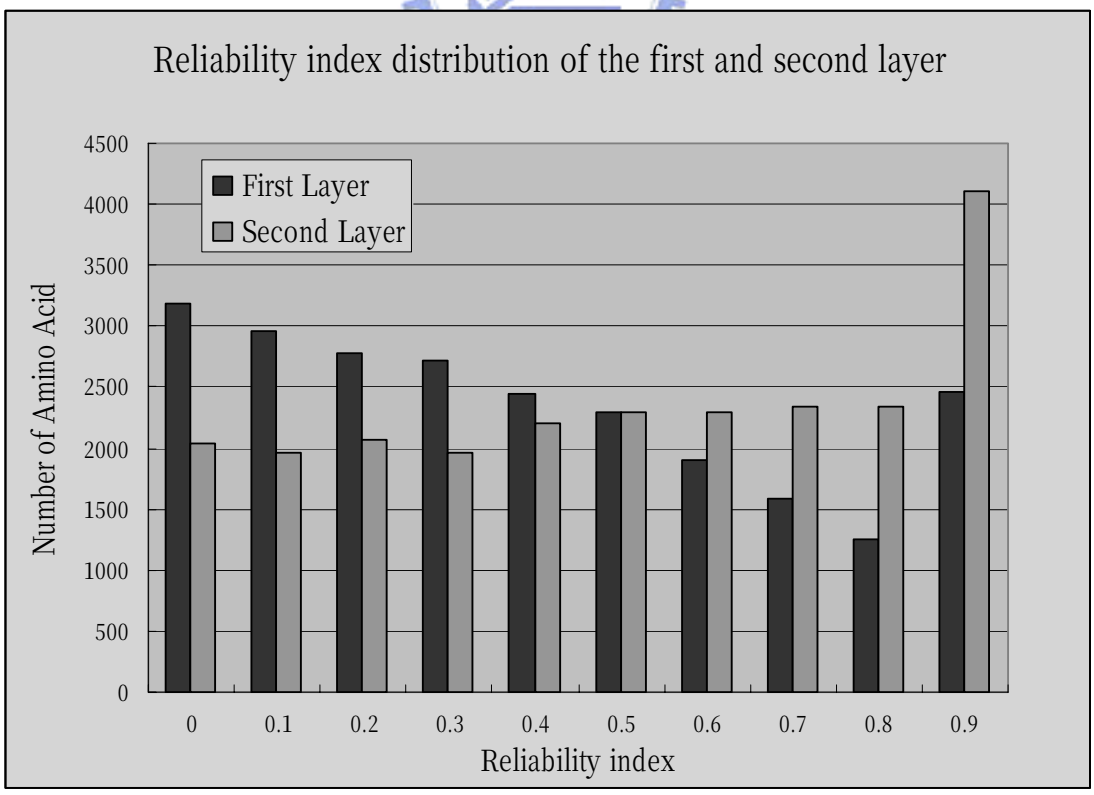


Fig. 3.6 The number distribution on different Reliability indices (from 0 to 0.9).

3.6.2 Linear Combination and Weighted Sum Fusion

Since we have proposed the dual-layer QuickRBF approach based on the PSSM profiles to predict the protein secondary structure, we found the difference of predictive results between the first and second layer are somewhat distinct. This motivates us to design a fusion scheme combining the results of each layer in order to raise the overall accuracy. The idea that we have hit upon is the linear combination scheme. Referring to Table 3.2, each residue of the present sequence has three target labels denoted as **C**, **E** and **H**. The respective results of each label are the linear combination of results of the first layer and second layer. Then the final output class of each residue is assigned to the one with the largest output value. Namely, it applies

$$P_{1j}(x) = \arg\max_{lc} f_{lc}(x) \qquad (lc \in \{C, E, H\}, j = 1,...,m) \tag{3.1}$$

where f_{lc} are the linearly combined values shown as follows:

$$\begin{aligned} f_{\mathrm{C}} &= C_{fj} = C_{1j} + C_{2j} \\ f_{\mathrm{E}} &= E_{fj} = E_{1j} + E_{2j} \\ f_{\mathrm{H}} &= H_{fj} = H_{1j} + H_{2j} \end{aligned} \tag{3.2}$$

The next fusion method involves the weighted sum scheme. Referring to Table 3.3, the respective results of three target labels are taken from the weighted sum of the first layer and second layer. The final output class of each residue is assigned to the one with the largest output value. Namely, it applies

$$P_{2j}(x) = \arg\max_{ws} f_{ws}(x) \qquad (ws \in \{C, E, H\}, j = 1,...,m) \tag{3.3}$$

where f_{ws} are the weighted sum shown as follows:

$$
\begin{aligned}
f_{\mathrm{C}} &= C_{fj} = w_1 C_{1j} + w_2 C_{2j} \\
f_{\mathrm{E}} &= E_{fj} = w_1 E_{1j} + w_2 E_{2j} \\
f_{\mathrm{H}} &= H_{fj} = w_1 H_{1j} + w_2 H_{2j}
\end{aligned} \tag{3.4}
$$

Table 3.2. Fusion method 1: Linear combination.

	First-Layer QuickRBF			Second-Layer QuickRBF			Fusion Method 2 (Linear Combination)			Pred 2
	Coil	**Sheet**	**Helix**	**Coil**	**Sheet**	**Helix**	**Coil**	**Sheet**	**Helix**	**Class**
1	C_{11}	E_{11}	H_{11}	C_{21}	E_{21}	H_{21}	C_{f1}	E_{f1}	H_{f1}	P_{11}
2	C_{12}	E_{12}	H_{12}	C_{22}	E_{22}	H_{22}	C_{f2}	E_{f2}	H_{f2}	P_{12}
3	C_{13}	E_{13}	H_{13}	C_{23}	E_{23}	H_{23}	C_{f3}	E_{f3}	H_{f3}	P_{13}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
m	C_{1m}	E_{1m}	H_{1m}	C_{2m}	E_{2m}	H_{2m}	C_{fm}	E_{fm}	H_{fm}	P_{1m}

Table 3.3. Fusion method 2: Weighted Sum.

	First-Layer QuickRBF			Second-Layer QuickRBF			Fusion Method 3 (Weighted Sum)			Pred 3
	Coil	**Sheet**	**Helix**	**Coil**	**Sheet**	**Helix**	**Coil**	**Sheet**	**Helix**	**Class**
1	C_{11}	E_{11}	H_{11}	C_{21}	E_{21}	H_{21}	C_{f1}	E_{f1}	H_{f1}	P_{21}
2	C_{12}	E_{12}	H_{12}	C_{22}	E_{22}	H_{22}	C_{f2}	E_{f2}	H_{f2}	P_{22}
3	C_{13}	E_{13}	H_{13}	C_{23}	E_{23}	H_{23}	C_{f3}	E_{f3}	H_{f3}	P_{23}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
m	C_{1m}	E_{1m}	H_{1m}	C_{2m}	E_{2m}	H_{2m}	C_{fm}	E_{fm}	H_{fm}	P_{2m}

Chapter 4. Experiment and Results

4.1 Datasets

The set 126 nonhomologous globular protein chains used in the experiment of Rost and Sander [1], referred to as the RS126 set, was used to evaluate the accuracy of the classifiers. The dataset contained 23606 residues with 32% α-helix, 23% β-strand, and 45% coil. Many current secondary structure prediction methods have been developed and tested on this dataset. The single stage approaches and second-stage approaches were implemented, with multiple sequence alignments, and tested on the dataset, using a sevenfold cross validation technique to estimate the prediction accuracy. With sevenfold cross validation approximately six-seventh of the database was selected for training and, after training, the left one-seventh of the dataset was used for testing. In order to avoid the selection of extremely biased partitions, the RS126 set was divided into seven subsets with each subset having similar size and content of each type of secondary structure as shown in Table 4.1. Referring to Table 4.1, four membrane proteins in the set G are eliminated in this thesis.

Table 4.1. The database of non-homologous proteins used for seven-fold cross validation. All proteins have less than 25% pairwise similarity for lengths great than 80 residues.

set A	256b_A	2aat	8abp	6acn	1acx	8adh	3ait
	2ak3_A	2alp	9api_A	9api_B	1azu	1cyo	1bbp_A
	1bds	1bmv_1	1bmv_2	3blm	4bp2		
set B	2cab	7cat_A	1cbh	1cc5	2ccy_A	1cdh	1cdt_A
	3cla	3cln	4cms	4cpa_I	6cpa	6cpp	4cpv
	1crn	1cse_I	6cts	2cyp	5cyt_R		
set C	1eca	6dfr	3ebx	5er2_E	1etu	1fc2_C	1fdl_H
	1dur	1fkf	1fnd	2fxb	1fxi_A	2fox	1g6n_A
	2gbp	1a45	1gd1_O	2gls_A	2gn5		
set D	1gpl_A	4gr1	1hip	6hir	3hmg_A	3hmg_B	2hmz_A
	5hvp_A	2i1b	3icb	7icd	1il8_A	9ins_B	1l58
	1lap	5ldh	1gdj	2lhb	1lmb_3		
set E	2ltn_A	2ltn_B	5lyz	1mcp_L	2mev_4	2or1_L	1ovo_A
	1paz	9pap	2pcy	4pfk	3pgm	2phh	1pyp
	1r09_2	2pab_A	2mhu	1mrt	1ppt		
set F	1rbp	1rhd	4rhv_1	4rhv_3	4rhv_4	3rnt	7rsa
	2rsp_A	4rxn	1s01	3sdh_A	4sgb_I	1sh1	2sns
	2sod_B	2stv	2tgp_I	1tgs_I	3tim_A		
set G	6tmn_E	2tmv_P	1tnf_A	4ts1_A	1ubq	2utg_A	9wga_A
	2wrp_R	1bks_A	1bks_B	4xia_A	2tsc_A	1prc_C	1prc_H
	1prc_L	1prc_M					

4.2 Results

4.2.1 Results of the Cascade of Fuzzy ARTMAP and QuickRBF

For the architecture of the cascade of fuzzy ARTMAP and QuickRBF, we choose a window size of 15 amino acid residues as input according to other benchmark researches such as PSIPRED [14] and SVM [4]. The empirical numbers of categories are about 12000, 10000 and 5000 while the defaulting ρ_a = 0.8, 0.7 and 0.4, respectively. We choose ρ_b = 1 since the encoded classes are linearly independent and orthogonal. We let the map field vigilance parameter ρ_{ab} be larger

than 0 which means that only the representative vectors holding the same classes will have the chance being learned. The $\beta = 1$ which means fast learning is adopted in this thesis.

Table 4.2 shows the performance of the secondary structure predictors using single QuickRBF and cascade of fuzzy ARTMAP (FARTMAP) and QuickRBF on the RS126 set with multiple sequence alignments. The multi-class techniques of the single QuickRBF gave the result for PSS prediction which achieved 75.75% of Q_3 accuracy while the cascade of the fuzzy ARTMAP and QuickRBF only achieved 73.91%, or even lower.

Table 4.2. Single QuickRBF and cascade of FARTMAP and QuickRBF.

RS126	QuickRBF	Cascade of FARTMAP & QuickRBF	Cascade of FARTMAP & QuickRBF	Cascade of FARTMAP & QuickRBF
Centers	5000	12000(ρ_a =0.8)	10000(ρ_a =0.7)	5000(ρ_a =0.4)
Set A	75.97	72.25	70.57	68.84
Set B	78.34	76.21	74.89	73.77
Set C	77.01	75.84	73.17	72.15
Set D	72.51	71.35	68.55	67.57
Set E	77.83	75.40	74.24	73.58
Set F	73.26	72.18	69.81	68.22
Set G	76.31	74.44	72.32	71.66
Average	75.75	73.91	71.92	70.77

4.2.2 Results of the Dual-Layer QuickRBF and Fusion Methods

For QuickRBF classifiers at the first stage, similarly we choose a window size of 15 amino acid residues as input. At the second stage, the window size of width 13 is

used as the coding scheme for second-layer QuickRBF technique. The numbers of center selected here were 5000 and 10000 on RS126 which we found the more the numbers of center the better the average accuracy. The first fusion method, linear combination of first and second layers, is determined empirically for optimal performance though the results are only slightly different from the second one. We have used several measures to evaluate the prediction accuracy. The Q3 accuracy indicates the percentage of correctly predicted residues of three states of secondary structure. The Q_C, Q_E, Q_H accuracies represent the percentage of correctly predicted residues of each type of secondary structure.

Tables 4.3 and 4.4 show the performance of the different secondary structure predictors using dual-layer QuickRBF on the RS126 set with multiple sequence alignments. Referring to Table 4.3, the multi-class techniques of first fusion method has an improved accuracy comparing to the results of each layer. It gave the result for PSS prediction which achieved 76.32% of Q_3 accuracy while the accuracy of the second fusion method was 76.31%. The best result was found to be the first fusion method using reliability index which achieved 76.71% of Q_3 accuracy while the prediction accuracy obtained by various methods are shown in Table 4.5 for comparison.

Table 4.3. Results of different fusion method with 5000 centers at first stage.

RS126	First layer QuickRBF	Second layer QuickRBF	Fusion Method 1 (RI)	Fusion Method 2 (Linear Combine)	Fusion Method 3 (Weighted sum)
Centers	5000	1000			
Set A	74.97	74.62	75.26	75.21	75.23
Set B	78.34	78.28	78.31	78.58	78.55
Set C	77.01	76.77	76.34	77.04	77.08
Set D	72.51	71.92	77.14	72.42	72.39
Set E	77.83	78.54	77.84	79.02	79.01
Set F	73.26	73.31	75.81	75.10	75.05
Set G	76.31	77.16	75.80	77.48	77.40
Average	75.75	75.80	76.42	76.32	76.31

Table 4.4. Results of different fusion method with 10000 centers at first stage.

RS126	First layer QuickRBF	Second layer QuickRBF	Fusion Method 1 (RI)	Fusion Method 2 (Linear Combine)	Fusion Method 3 (Weighted sum)
Centers	10000	1000			
Set A	75.62	73.91	74.99	75.14	75.16
Set B	78.37	78.46	79.19	79.16	79.10
Set C	77.31	76.98	76.57	78.09	78.07
Set D	72.42	73.44	77.67	73.49	73.45
Set E	77.97	78.25	78.51	78.35	78.41
Set F	74.27	74.35	76.06	75.98	75.94
Set G	76.20	76.43	75.86	76.75	76.72
Average	76.02	75.97	76.71	76.61	76.60

Referring to Table 4.5, PHD [2]: results obtained from Rost and Sander. DSC, PREDATOR, NNSSP, CONSENSUS [7]: results obtained on the RS126 set from Cuff and Barton. PMSVM [12]: results obtained on CB513 data set from Guo *et al.*. SVMpsi [23]: results obtaind on RS126 data set from Kim and Park. QRBF [5]: results obtained from the QuickRBF proposed by Ou *et al.*. dQRBF: results obtained from dual-layer QuickRBF. dfQRBF: results obtained from the first fusion method combining QRBF and dQRBF.

Table 4.5. Results comparison of several methods obtained on RS126 dataset.

Method	Q_3 (%)	Q_C (%)	Q_E (%)	Q_H (%)
PHD [2]	70.8	72.0	66.0	72.0
DSC [7]	71.1	—	—	—
PREDATOR [7]	70.3	—	—	—
NNSSP [7]	72.7	—	—	—
CONCENSUS [7]	74.8	—	—	—
PMSVM [12]	75.2	72.8	71.5	80.4
SVMpsi [23]	76.1	77.2	63.9	81.5
QRBF [5]	76.0	75.0	63.1	82.9
dQRBF	76.0	77.3	64.3	80.9
dfQRBF	76.7	76.8	64.5	84.7

Chapter 5. Conclusion

In this thesis, a dual-layer QuickRBF is proposed which raises the accuracy to 76.7% at best. Besides, we also intend to improve the performance of QuickRBF via modifying the QuickRBF architecture by using fuzzy ARTMAP for center generation. According to the results, it seems that modified architectures of QuickRBF fail to raise the accuracy. However, it is demonstrative that the dual-layer QuickRBF makes a breakthrough in improving final prediction accuracy which achieves 76.7% in terms of Q3. With the aids of fusion method, we think this is a promising way combining the results, i.e. 3-state regression value, of different approaches such as SVM (support vector machine). We believe that the combination of QuickRBF and SVM would bring the current Q3 accuracy into a new high level of reliability due to their respective advantages of algorithm. Combined results of QuickRBF and SVM are now in the working in our lab and we hope that it will reach a higher accuracy on the RS126 dataset. It is suggested that the combination of regression value of different approaches will achieve higher prediction accuracies. Besides, a drawback of the neural network approach is that, it is unclear how the additional evolutionary information affects the prediction accuracy. The inside of a learned neural network approach is hard to understand and to translate into useful knowledge. Therefore, the second future work is to further study the noise reduction existing in the dataset fed into the neural network. If the individual entities of training dataset are not independent and representative, it introduces a glut of interference which results in poor performance. Some of these aspects are introduced by Guo *et al.* [12]. Therefore, practical coded scheme may be developed to encompass different informative profiles and possibly filter the potential noise off.

References

[1] B. Rost, and C. Sander, “Prediction of protein secondary structure at better than 70% accuracy,” *J. Mol. Biol.*, 232, 584–599, 1993.

[2] B. Rost, and C. Sander, “Combining evolutionary information and neural networks to predict protein secondary structure,” *Proteins*, 19, 55–72, 1994.

[3] B. Rost, “Review: Protein secondary structure prediction continues to rise,” *J. Struct. Biol.* 134, 204–218, 2001.

[4] S. Hua and Z. Sun, “A novel method of protein secondary structure prediction with high segment overlap measure: support vector machine approach,” *J. Mol. Biol.*, 308:397–407, 2001.

[5] Y. Y. Ou, “QuickRBF: an efficient RBFN package,” Software available at *http://csie.org/~yien/quickrbf*, 2005.

[6] C. C. Chang and C. J. Lin, “LIBSVM: a library for support vector machines,” Software available at *http://www.csie.ntu.edu.tw/~cjlin/libsvm*, 2001

[7] J. A. Cuff and G. J. Barton, “Evaluation and improvement of multiple sequence methods for protein secondary structure prediction,” *Proteins*: 34:508–519, 1999.

[8] K. P. Wu, H. N. Lin, J. M. Chang, T. Y. Sung, and W. L. Hsu, “HYPROSP: a hybrid protein secondary structure prediction algorithm knowledge-based approach,” *Nucleic Acids Research*, 32(17):5059–5065, 2004.

[9] H. N. Lin, J. M. Chang, K. P. Wu, T. Y. Sung, and W. L. Hsu, “HYPROSP II: A knowledge-based hybrid method for protein secondary structure prediction based on local prediction confidence,” *Bioinformatics*, 21:3227–3233, 2005.

[10] M. Georgiopoulos, H. Fernlund, G. Bebis, and G.. Heileman, “Order of search in fuzzy art and fuzzy ARTMAP: Effect of the choice parameter,” *Neural Networks*,

vol. 9, no. 9, pp. 1541–1559, 1996.

[11] G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds, and D. B. Rosen, “Fuzzy ARTMAP: A neural network architecture for incremental learning of analog multidimensional maps,” *IEEE Transactions on Neural Networks*, 3(5), 698–713, 1992 .

[12] J. Guo, H. Chen, Z. Sun, and Y. Lin, “A novel method for protein secondary structure prediction using dual-layer SVM and profiles,” *Proteins*, 54:738–743, 2004.

[13] S. K. Riis and A. Krogh, “Improving prediction of protein secondary structure using structured neural networks and multiple sequence alignments,” *J. Comput. Biol.*, 1:163–183, 1996.

[14] D. T. Jones, “Protein Secondary Structure Prediction Based on Position-specific Scoring Matrices,” *J. Mol. Biol.*, vol. 292, pp. 195–202, 1999.

[15] E. Parrado-Hernández, E. Gómez-Sánchez, and Y. A. Dimitriadis, “Study of distributed learning as a solution to category proliferation in Fuzzy ARTMAP based neural systems,” *Neural Networks*, 16, 1039–1057, 2003.

[16] E. Gómez-Sánchez, Y. A. Dimitriadis, J. M. Cano-Izquierdo, and J. López-Coronado, “ μARTMAP: Use of mutual information for category reduction in fuzzy ARTMAP,” *IEEE Transcations on Neural Networks*, 23(1), 58–69, 2002.

[17] S. J. Verzi, G. L. Heileman, M. Georgiopoulos, and M. J. Healy, “Boosted ARTMAP,” *IEEE World Congr. Comput. Intell.*, pp. 396–400, 1998.

[18] J. Castro, M. Georgiopoulos, R. Demara, A. Gonzalez, “Data-Partitioning using the Hilbert Space Filling Curves: Effect on the speed of convergence of fuzzy ARTMAP for large database problems,” *Neural Networks*, 18(7), 967–984, 2005.

[19] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. H. Zhang, Z. Zhang, W. Miller,

and D. J. Lipman, “Gapped BLAST and PSI-BLAST: A new generation of protein database search programs,” *Nucleic Acids Research*, vol. 25, pp. 3389–3402, 1997.

[20] T. N. Petersen, C. Lundegaard, M. Nielsen, H. Bohr, and J. Bohr, “Prediction of protein secondary structure at 80% accuracy,” *Proteins*, 41, 17–20, 2000.

[21] J. Selbig, T. Mevissen, and T. Lengauer, “Decision tree-based formation of consensus protein secondary structure prediction,” *Bioinformatics*, 15, 1039–1046, 1999.

[22] M. N. Nguyen and J. C. Rajapakse, “Multi-class support vector machines for protein secondary structure prediction,” *Genome Informatics*, 14, 218–227, 2003.

[23] H. Kim and H. Park, “Protein secondary structure prediction based on an improved support vector machines approach,” *Protein Engineering*, 16(8), 553–560, 2003.

[24] G. A. Carpenter, B. L. Milenova, and B. W. Noeske, “Distributed ARTMAP: A neural network for fast distributed supervised learning,” *Neural Networks*, 11(2), 323–336, 1998.

[25] P. Tomaso and G. Federico, *A theory of networks for approximation and learning*, MIT Artificial Intelligence Laboratory and Center for Biological Information Processing, Whitaker College, 1989.