2009 International Conference of Soft Computing and Pattern Recognition

Protein Fold Pattern Recognition Using Bayesian Ensemble of RBF Neural Networks

Homa Baradaran Hashemi, Azadeh Shakery, Mahdi Pakdaman Naeini
Department of Electrical and Computer Engineering
University of Tehran
Tehran, Iran
e-mail: {H.B.Hashemi, Shakery, m.pakdaman}@ece.ut.ac.ir

Abstract—**Protein fold pattern recognition has been one of the most challenging problems in biology during the last 40 years. Recently due to the vast improvement in machine learning and pattern recognition methods many computer scientists have applied these methods to solve this problem. However, protein folding problem is much more complicated than ordinary machine learning problems because of its natural complexity imposed by the high dimensionality of feature space and diversity of different protein fold classes. To deal with such a challenging problem, we use an ensemble classifier model by applying MLP and RBF Neural Networks and Bayesian ensemble method. Also we have used the Laplace estimation method in order to smooth confusion matrices of the base classifiers. Experimental results imply that RBF Neural Network holds better Correct Classification Rate (CCR) compared to other common classification methods such as MLP networks. Our experiments also show that the Bayesian fusion method can improve the correct classification rate of proteins up to 20% with the final CCR of 59% by reducing both bias and variance error of the RBF classifiers, on a benchmark dataset containing 27 SCOP folds.**

Keywords-Protein Folding; Bayesian Classifier Fusion; RBF; MLP

I. Introduction

Proteins are large biological macromolecules which organize essential parts of living organisms to control all of their vital functionalities. Protein functions are related to protein chemical reactions with their surrounding and other proteins. Also, protein functions depend on its shape and three-dimensional (3D) structure. The protein folding is the process by which the protein assumes its characteristic 3D structure after the translation process in a cell. Incorrectly folded proteins usually cause to produce inactive proteins with different properties which are believed to be the result of some diseases. Consequently, being aware of the correct 3D structure of many proteins is an essential problem in biology. Since determining the 3D structure of a protein by experiment is a very difficult and expensive process, scientists have tried to model protein folding phenomenon by using different biophysical techniques.

Some early works on predicting protein structural classes include supervised fuzzy clustering approach [24], amino acid index [7], amino acid principal component analysis [15], amino acid distributions [20], Bayesian classifier [9], discriminant analysis [19], hydrophobicity profiles [17] and correlation coefficient [11].

Recently, due to the vast improvements in computers' power, computer scientists have become interested in the protein folding problem using the machine learning and pattern recognition methods [13, 3, 25, 1, 16]. However the ordinary and common classification methods do not work very well on this problem due to high dimensional feature space and multiple classes [25, 23]. In this work, we use a taxonomic approach similar to the methods developed by Ding and Dubchuk (2001) [13] and Shen and Chou (2006) [25]. In this approach number of protein folds is assumed to be restricted, so predicting the 3D structure can be converted into fold classification problem.

In this paper, we applied two classification methods: MLP and RBF networks. Also we used Bayesian and Majority Voting classifier ensemble methods to improve the prediction results of the base classifiers. In the following, we briefly introduce artificial Neural Networks and the ensemble methods used in our study in section II. In section III we introduce the dataset properties and data preprocessing. We also present our experimental results in comparison with the previous work. Finally we bring the conclusions and future work of our study in section IV.

II. Methods

Inductive learning methods are categorized as supervised and unsupervised methods. In this section we introduce both a brief introduction to the supervised classification and classifier ensemble methods which are used in this study; more details can be found in [5].

A. Artificial Neural Networks

The neural network is a very applicable regression and classification tool which has the capability of representing complex relationships among inputs and outputs of a system. The important advantage of neural networks lies in their ability to be a general function approximator and learn both the linear and complex nonlinear relationships directly from the data. Intuitively, neural networks imitate the human brain intelligent behavior using a connectionism approach. In this view, an artificial neural network is constructed from a set of connected simple computational units known as neurons. Each neuron does two important jobs: (1) computing the

978-0-7695-3879-2/09 $26.00 © 2009 IEEE
DOI 10.1109/SoCPaR.2009.91

weighted sums of its inputs; and (2) using a non-linear mapping on the results of the previous stage.

The most common neural network models are the Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF) networks which are used in this paper. In MLP structure, the hyperbolic tangent or sigmoid functions are used as the nonlinear transfer functions of the hidden nodes and also Back propagation error learning algorithm is used for adjusting the weights. RBF networks have a static Gaussian function as the nonlinearity for the hidden layer's neurons and there are different methods for training RBF networks.

B. Ensemble Methods

The main idea of classifier ensemble methods is to acquire better classification results by fusing the outcomes of some base classifiers. Different research studies that have approached challenging biological problems using various machine learning techniques show that usually the best performance is obtained by different methods, which indeed justifies the No Free Lunch theorem. As a result, it seems that by using classifier fusion methods and reducing the bias and variance error of the base classifiers we can improve the final classification precision.

There are distinctive ensemble methods which are used in all levels of information fusion: data, features and decision levels. In this paper we applied ensemble methods on the decision level of some basic classifier outputs to improve the classification results. In the following we describe briefly the ensemble methods used in our study.

1) Majority Voting: Majority voting method is one of oldest, most popular and effective decision making strategies. This method is used in the decision level of fusion task in which the vote of the majority will be accepted as the final output of the classifier.

2) Bayesian Fusion: Bayesian approach is another popular and effective classifier fusion method. In this method the normalized confusion matrix of each base classifier is used to ensemble their outputs. Assume that we have N different classifiers and each classifier works independently. In addition, we assume that each instance should be classified in one of the target classes $C_1, C_2, \dots, C_k$. Considering these assumptions, if $X_1, X_2, \dots, X_N$ indicate the decision of N different base classifiers about the actual class of a particular instance, the final optimal class of the instance would be the one which has the maximum posterior probability contingency to the observations $X_1, X_2, \dots, X_N$. We can formulate this process as following:

$$C_{\mathrm{opt}} = \underset{j=1}{\overset{k}{argMax}}\ \mathrm{P}(\mathrm{C_j}|\mathrm{X_1}, \mathrm{X_2}, \dots, \mathrm{X_N})$$

Using the independence assumption on $\mathrm{X_1}, \mathrm{X_2}, \dots, \mathrm{X_N}$ and the Bayes probability theory, the posterior probability of the final class can be calculated as below:

$$\mathrm{P}(\mathrm{C_k}|\mathrm{X_1}, \mathrm{X_2}, \dots, \mathrm{X_N}) = \prod_{i=1}^{N} \mathrm{P}(\mathrm{C_k}|\mathrm{X_i}) = \prod_{i=1}^{N} \frac{\mathrm{P}(\mathrm{X_i}|\mathrm{C_k})\mathrm{P}(\mathrm{C_k})}{\mathrm{P}(\mathrm{X_i})}$$

By using the log likelihood function of the posterior probability and omitting the denominator of the obtained expression we can calculate the optimal Bayes estimated class as below:

$$\mathrm{C_{opt}} = \underset{j=1}{\overset{k}{\mathrm{argMax}}} \left(\sum_{i=1}^{N} \log \mathrm{P}(\mathrm{X_i}|\mathrm{C_j}) + \mathrm{N} \log \mathrm{P}(\mathrm{C_j}) \right)$$

In the above expression, there are usually two different ways of finding the prior probability distribution of classes, $\mathrm{P}(\mathrm{C_j})$ .We can either use a uniform distribution or the distribution of the classes on training dataset. In our experiments we have used the later solution to estimate $\mathrm{P}(\mathrm{C_j})$. In order to find the value of $\mathrm{p}(\mathrm{X_i}|\mathrm{C_j})$ we can use the normalized confusion matrix of the base classifiers. However, since there are many zero terms in this matrix we have to smooth these items. We use Laplace estimate for smoothing purpose in our experiments. In the Laplace method, if the prior probability estimate of the random variable X is equal to $\mathrm{P_{prior}}(\mathrm{X}) = \frac{\mathrm{m_0}}{\mathrm{n_0}}$ and the posterior probability of X is equal to $\mathrm{P_{posterior}}(\mathrm{X}) = \frac{\mathrm{m_1}}{\mathrm{n_1}}$ then the smoothed probability of X would be $\mathrm{P_{Smoothed}}(\mathrm{X}) = \frac{\mathrm{m_1} + \alpha \mathrm{m_0}}{\mathrm{n_1} + \alpha \mathrm{n_0}}$ where α is a constant that shows relative degree of confidence to the prior knowledge. In the classification task it can be determined by examining the performance of ensemble classifier on the validation dataset. Also, in this experiment we use the value of $\frac{1}{\mathrm{K}}$ as the prior probability for $\mathrm{P_{prior}}(\mathrm{X_i}|\mathrm{C_j})$.

III. Experiments and Results

We have done some experiments to evaluate the performance of our proposed method. In this section, we present the experimental results of our work on the protein folding problem.

A. The Dataset

The dataset in this study has been obtained from Ding and Dubchak (2001) [18] which has been a very popular dataset [25, 12, 4, 27, 21, 26, 8, 22]. The original training dataset and testing dataset respectively contain 313 and 385 proteins. Due to lack of information on sequence records of two proteins (2SCMC and 2GPS) in the training dataset and two proteins (2YHX_2 and 2YHX_1) in testing dataset, we excluded these four proteins from the working dataset. Consequently, in our experiments we used 311 proteins for training and 383 proteins for testing. In this dataset each two proteins have no more than 35% of sequence identity for the aligned subsequences longer than 80 residues. Also there are 27 different protein folds in this dataset in which each fold has at least seven proteins [13].

Considering the structural class, among these limited fold types, 6 types belong to all α structural class, 9 types to all β class, 9 types to α / β class and 3 types to α + β class.

Ding and Dubchak in their work [13] extracted the following six features from sequence of proteins [13]: Amino acids composition, Predicted secondary structure, Hydrophobicity, Normalized van der Waals volume, Polarity and Polarizability.

TABLE I. SIX EXTRACTED FEATURES WITH THEIR DIMENSIONS FROM PROTEIN SEQUENCE

Dataset Feature	*Dimension*
Amino acids composition	20
Predicted secondary structure	21
Hydrophobicity	21
Normalized van der Waals volume	21
Polarity	21
Polarizability	21

Of the above six features, only the Amino acids composition contains 20 components (number of native amino acids) for each protein. Each of the other five features contains 21 components. Table I shows extracted parameters from protein sequence. Accordingly, there is a 125 dimensional feature vector for each protein in this dataset.

B. Experimental Results

Since the problem of classification in this case study is a multi class prediction problem, we use 27 different output units in the structure of RBF and MLP networks. Also, in our study the MLP network has just one hidden layer with tangent sigmoid as activation function. For recognizing the exact class of a protein we use the label of the maximum output unit in the network as the protein class label. Moreover, we use Correct Classification Rate (CCR) as the evaluation measure which is the number of correct classified instances over the total number of instances.

In order to find the optimum value of the constant α in the Laplace estimate smoothing method, we have used the CCR of ensemble model on the validation dataset. In this approach the point in which we have got the minimum error is selected as the optimum value of α. Fig. 1 and 2 show the CCR of different methods on the test dataset.

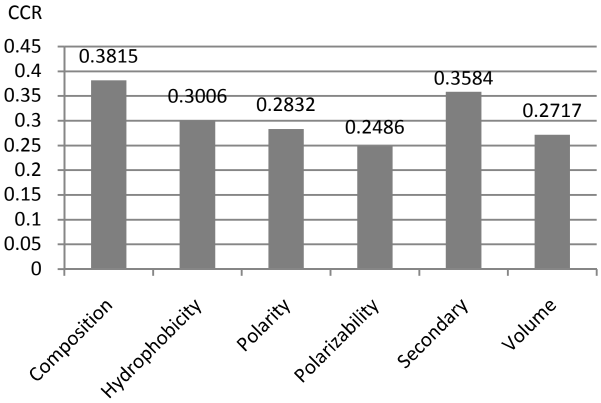

Figure 1. Correct Classification rate of MLP network.

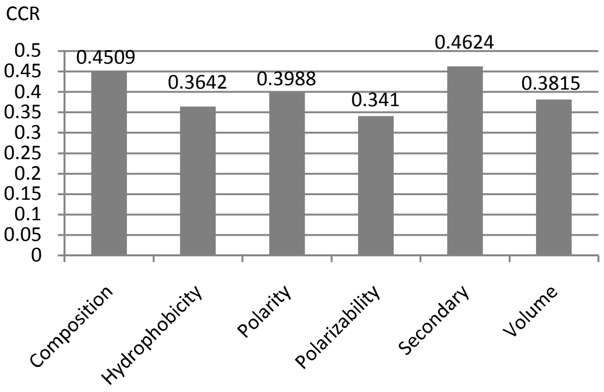

Figure 2. Correct Classification rate of RBF network.

Moreover, by using classifier ensemble methods such as Bayesian classifier we can improve the correct classification rate of proteins up to 20% and the final Correct Classification Rate becomes around 59%. The results are shown in Fig. 3 and 4. It shows that the classifier ensemble works better than the other methods, which indeed justifies No Free Lunch theorem. This improvement can be described by the concept of Bias and Variance error of a learner which we discuss in the following section.

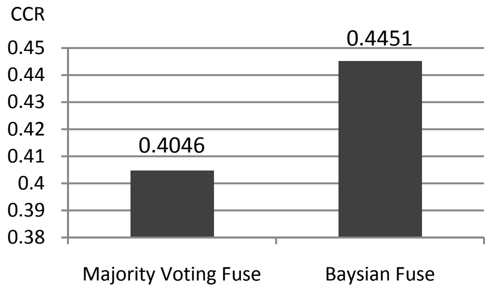

Figure 3. Ensemble CCR of MLP network.

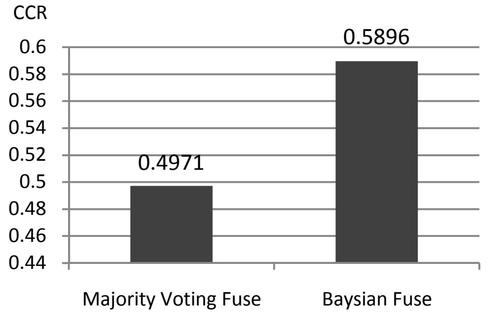

Figure 4. Ensemble CCR of RBF network.

C. *Discussion*

In this section we justify the obtained improvements in protein fold recognition using the Bayesian ensemble method by describing the bias and variance error of a classifier. This is one of the basic concepts in learning theory which can also justify the cause of beating a simple learner against a complex one. Moreover, it can be used to prevent a learner being over fitted to the training data.

In a learning problem, assume that $S = \{(x_1, t_1), \dots, (x_n, t_n)\}$ is the set of all training data and the learner L is going to learn the concept y=f(x). In order to evaluate the quality of the learner, we usually use a Loss function that can be declared in different ways such as: Zero-One Loss, Squared Loss or Absolute Loss function. Generally, a Loss function can be written as the summation of bias, variance and the noise [14, 5].

Let's assume that we want to learn a quadratic function using a linear learner L and we have many different training sets of the target function. By using different training sets of the quadratic function we will obtain different lines e.g. different least squared lines and if we average these lines together we will obtain another line. Since our target function is a quadratic one and we want to learn it by a linear hypothesis, the value of estimated function with the average line, for every point, will have some error with the true target value of that point. We call this type of estimate error as bias error of the learner.

Bias error of learner exists since hypothesizes do not have the ability of showing entirely the true target concept. The bias error in the linear regression of a quadratic function is shown in Fig. 5 (a).

Consequently, if the learner is a general function approximator by averaging out the result of learning over all different training data we can somewhat cancel out the bias error of our learner. However, the loss function would not be zero because of the variance error of the learner. In Fig. 5 (b) we assume that the points labeled by x, o, s are three different training sets of the true quadratic target function showed by the solid black curve. Since we have some noise in our measurements they are not exactly on the curve.

Suppose for each training set we use any general function approximator and the learning process gives us three different dashed curves completely learned on training data. The variance error for each learner on every point is the difference between the estimated target value at that point and the average of the estimates over all learners. This concept is shown in the Fig. 5 (b).

We cannot remove completely both the variance and bias error of a classifier together and there is an optimum point in between. For example although the bias error of an MLP is less than the bias error of a single perceptron, there are some cases in which the variance error of a single perceptron is much less that the variance error of an MLP. Consequently, sometimes we see that a single perceptron can beat a complex MLP [14, 5].

By using different classifier ensemble methods such as bagging, boosting or Bayesian method, we can usually decrease both the bias and variance error of a learner. For example when we use bagging we are simulating the case when we have many different training sets and by using the majority votes between the final classifiers we are decreasing the variance error of the classification problem. On the other hand, by using more advanced ensemble methods such as boosting or Bayesian ensemble method, we can decrease both bias and variance error of the learner [23, 10, 6, 2].

D. *Comparison with the other fold recognition methods*

We compared the performance of our approach with three other fold recognition methods which are based on same training and testing datasets (Ding and Dubchak, 2001; Chung and Huang, 2003; Shen and Chou, 2006). Ding and Dubchak (2001) proposed a method based on support vector machines and neural networks as base classifiers and majority voting to combine scores of multiple parameter datasets. They reported overall success of 56% in predicting protein folds of testing dataset.

Chung and Huang (2003) proposed a novel hierarchical learning architecture which can be formed by neural networks and support vector machines as basic building blocks.

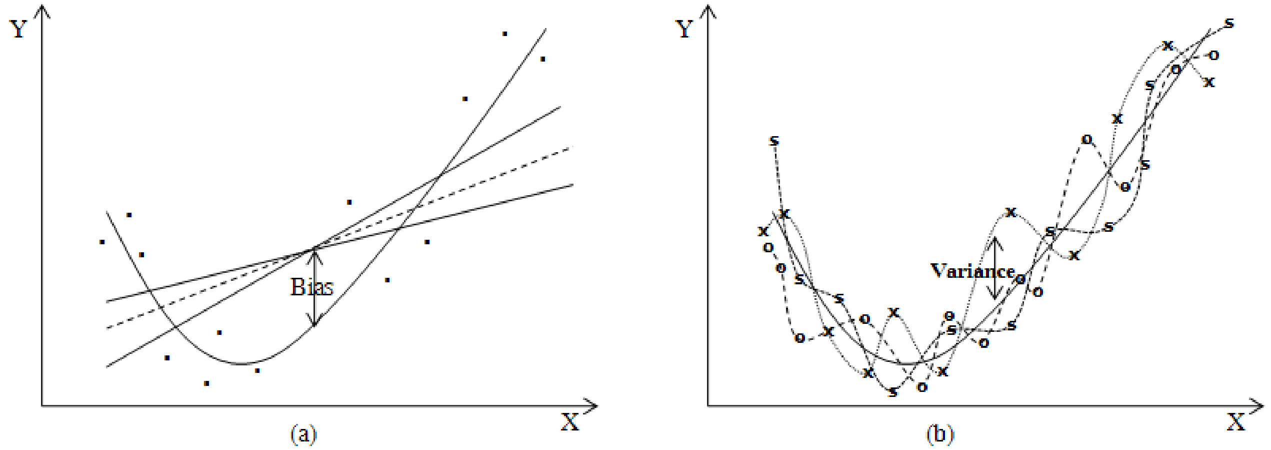

Figure 5. (a) Bias error of learning a quadratic function using a linear learner (b) Variance error of learning a quadratic function using a general function approximator.

As can be seen from TABLE II. , with the same set of features, our method obtains better overall correct classification rate in comparison with other methods.

Another recent research, Shen and Chou (2006), used OET-KNN (optimized evidence-theoretic k-nearest neighbors) and weighted majority voting method to ensemble classifiers outputs. They used three other extra features which are extracted from protein sequences. Success rate of this method is reported 62.5% over testing dataset. This experiment also shows that classifier ensemble is an appropriate approach to recognize protein fold. The main difference of Shen and Chou (2006) with our research is that they have used different orders of pseudo amino acid composition and structural properties of amino acids as features. Table II describes the correct classification rate of the proposed methods compared with some other methods. From this table it is clear that our result is remarkably better than the others in CCR with the same set of features.

IV. Conclusions and Future Work

In this paper, we studied the performance of classifier ensemble methods in the context of high dimensional multi class protein fold pattern recognition problem. We examined different base classifier such as MLP and RBF neural networks. Finally, due to the better generalization performance of RBF networks, we used six different RBF networks trained on six different feature sets of proteins extracted from their sequence information which are: Amino acids composition, Predicted secondary structure, Hydrophobicity, Normalized van der Waals volume, Polarity and Polarizability.

TABLE II. Classification results of related papers base on correct classification rate of different approaches

Classifier	*Reference*	*CCR (%)*
MLP[1]	[12]	48.8
GRNN[2]	[12]	44.2
RBFN[3]	[12]	49.4
NN[a]	[13]	41.8
SVM[b]	[13]	45.2
SVM[c]	[13]	51.1
SVM[d]	[13]	56.0
Ensemble Classifier[e]	[25]	62.1
MLP majority voting fuse	current research	40.46
MLP Bayesian fuse	current research	44.51
RBF majority voting fuse	current research	49.71
RBF Bayesian fuse	current research	**58.96**

a. The training method for NN is 'one against others'.
b. The training method for SVM is 'one against others'.
c. The training method for SVM is 'unique one against others'.
d. The training method for SVM is 'all against all'.
e. The ensemble classifier is constructed by nine OET-KNN classifiers and the number of neighbors in each OET-KNN classifier is 8.

Accordingly, we have totally a 125 dimensional feature vector for each protein, so this problem can be categorized as an ill-posed classification problem due to the curse of dimensionality phenomena.

Our experiments show that the use of Bayesian ensemble method is very promising in the problem of protein fold recognition because of decreasing both bias and variance error of the base classifiers in this approach. Also, we used Laplace estimate method in order to smooth the confusion matrix of the base classifiers to be prepared for fusion task. In our future work we are going to use other protein features extracted from protein sequence and structure such as pseudo-composition acid. Also we are going to examine other classification and classifier ensemble methods. Moreover, in order to decrease the high dimensionality of feature space we are going to use various feature transformation techniques such as PCA and ICA.

References

[1] P. Baldi and S. Brunak, "Bioinformatics: The Machine Learning Approach," Adaptive computation and machine learning, second ed. MIT press, 2001.

[2] E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: Bagging, Boosting and variants," Machine Learning, vol. 36, pp. 105-142, 1999.

[3] H. Bhaskar, D. C. Hoyle, and A. Singh, "Machine Learning in bioinformatics: A brief survey and recommendations for practitioners," Journal of computers in biology and Medicine-vol. 36, pp. 1104-1125, 2006.

[4] E. Bindewald, A. Cestaro, J. Hesser, M. Heiler, and S.C. Tosatto, "MANIFOLD: protein fold recognition based on secondary structure, sequence similarity and enzyme classification," Protein Eng, 16(11):785-789, 2003.

[5] C. M. Bishop, "Pattern Recognition and Machine learning," Second edition: Springer, 2006.

[6] L. Breiman, "Bagging Predictors," Machine Learning, vol. 24, no. 2, pp. 123-140, 1996.

[7] W. S. Bu, Z. P. Feng, Z. Zhang, and C. T. Zhang, "Prediction of protein (domain) structural classed based on amino-acid index," Eur. J. Biochem. 266, 1043– 1049, 1999.

[8] K. Chen and L. Kurgan, "PFRES: protein fold classification by using evolutionary information and predicted secondary structure," Bioinformatics, 23(21), 2843- 2850, 2007.

[9] A. Chinnasamy, W. K. Sung, and A. Mittal, "Protein structure and fold prediction using tree-augmented Bayesian classifier," Pacific Symp, Biocomput, vol. 9, pp. 387–398, 2004.

[10] S. Cho and J. Ryu, "Classifying Gene Expression Data of Cancer using Classifier Ensemble with Mutually Exclusive Feature," IEEE Proceeding, vol. 90, No. 11, 2002.

[11] K. C. Chou and C. T. Zhang, "Diagrammatization of codon usage in 339 human immunodeficiency virus proteins and its biological implication," AIDS Res. Hum. Retroviruses, vol. 8, pp. 1967–1976, 1992.

[12] I. F. Chung and C. D. Huang, "Recognition of structure classification of protein folding by NN and SVM hierarchical learning architecture," In Lecture Notes in Computer Sciences (Kaynak, O., Alpaydin, E., Oja, E. & Xu, L., eds.), vol. 2714, pp. 1159-1167. Springer, Istanbul, Turkey. a10, 2003.

[1] Multi Layer Perceptron neural network

[2] General Regression Neural Networks

[3] Radial Basis Function Network

[13] C. H. Ding and I. Dubchak, “Multi-class protein fold recognition using support vector machines and neural networks,” Bioinformatics, vol. 17, pp. 349–358.a3, 2001.

[14] P. Domingos, “A Unified Bias-Variance Decomposition for Zero-one and Squared Loss,” In Proc. of the 17th National Conference on Artificial Intelligence, pp. 564-569, 2000.

[15] Q. S. Du, Z. Q. Jiang, W. Z. He, D. P. Li, and K. C. Chou, “Amino acid principal component analysis (AAPCA) and its application in protein structural class prediction,” J. Biomol. Struct. Dynam. 23(6) 635–640, 2006.

[16] G. W. Greenwood, J. Shin, B. Lee, and G. B. Fogel, “A Survey of Recent Works on Evolutionary Approaches to the Protein Folding,” IEEE, pp. 488-495, 1999.

[17] M. M. Gromiha and P. K. Ponnuswamy, “Prediction of protein secondary structures from their hydrophobic characteristics,” Int. J. Pept. Protein Res. 45, 225–240, 1995.

[18] http://ranger.uta.edu/~chqding/protein

[19] P. Klein, “Prediction of protein structural class by discriminant analysis,” Biochim. Biophys. Acta 874, 205–215, 1986.

[20] T. S. Kumarevel, M. M. Gromiha, and M. M. Ponnuswamy, “Structural class prediction of residue distribution along the sequence,” Biophys Chem. 88, 81–101, 2000.

[21] F. Liang, “An effective Bayesian neural network classifier with a comparison study to support vector machine,” Neural Computation, 15:1959–1989, 2003.

[22] K. L. Lin, C. Y. Lin, C. D. Huang, H. M. Chang, C. T. Lin, C. Y. Tang, and D. F. Hsu, “Improving prediction accuracy for protein structure classification by neural networks using feature combination,” Proceedings of the 5th WSEAS International Conference on Applied Informatics and Communications (AIC’05), pp. 313-318, 2005.

[23] L. Nanni and A. Lumini, “Ensemblator: An ensemble of classifiers for reliable classification of biological data,” Journal of Pattern Recognition Letters, vol 28, pp. 622-630, 2007.

[24] H. B. Shen, J. Yang, X. J. Liu, and K. C. Chou, “Using supervised fuzzy clustering to predict protein structural classes,” Biochem. Biophys. Res. Commun. 334, 577–581, 2005.

[25] H. B. Shen and K. C. Chou, “Ensemble Classifier for protein fold pattern recognition,” Journal of Bioinformatics, vol 22, no. 14, pp. 1717-1722, 2006.

[26] A. C. Tan, D. Gilbert, and Y. Deville, “Integrative machine learning approach for multi-class SCOP protein fold classification,” Proceedings of the German Conference on Bioinformatics (GCB 2003), pp. 153–159, 2003.

[27] A. C. Tan, D. Gilbert, and Y. Deville, “Multi-class protein fold classification using a new ensemble machine learning approach,” Genome Informatics, vol.14, pp.206–217, 2003.