Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 1184 – 1196
© TÜBİTAK
doi:10.3906/elk-1506-126

# Protein fold classification with Grow-and-Learn network

**Özlem POLAT**[1,*], **Zümray DOKUR**[2]
[1]Department of Biomedical Engineering, Faculty of Technology, Cumhuriyet University, Sivas, Turkey
[2]Department of Electronics and Communication Engineering, Faculty of Electrical and Electronics Engineering, İstanbul Technical University, İstanbul, Turkey

**Received:** 14.06.2015 • **Accepted/Published Online:** 09.04.2016 • **Final Version:** 10.04.2017

**Abstract:** Protein fold classification is an important subject in computational biology and a compelling work from the point of machine learning. To deal with such a challenging problem, in this study, we propose a solution method for the classification of protein folds using Grow-and-Learn (GAL) neural network together with one-versus-others (OvO) method. To classify the most common 27 protein folds, 125 dimensional data, constituted by the physicochemical properties of amino acids, are used. The study is conducted on a database including 694 proteins: 311 of these proteins are used for training and 383 of them for testing. Overall, the classification system achieves 81.2% fold recognition accuracy on the test set, where most of the proteins have less than 25% sequence identity with the ones used during the training. To portray the capabilities of the GAL network among the other methods, comparisons between a few approaches have also been made, and GAL's accuracy is found to be higher than those of the existing methods for protein fold classification.

**Key words:** Protein fold classification, grow and learn neural network, attributes for protein fold recognition, bioinformatics

## 1. Introduction

Proteins are large biomolecules responsible for many vital functions within living organisms. A protein's functions depend on its shape and three-dimensional (3D) structure [1]. Computational analysis of biological data obtained in protein structure is essential for understanding protein function and the discovery of new drugs and therapies. There are currently 116,085 (at 18/02/2016) experimentally determined 3D structures of proteins deposited in the Protein Data Bank (http://www.rcsb.org) and the number of these structures increases day by day. However, there are many similar structures without sequence similarity in this protein set. Therefore, comparison of protein structures, fold classification, and fold recognition became topics of interest in computational biology.

Proteins can be classified into one of four structural classes based on their secondary structure components: all-$\alpha$, all-$\beta$, $\alpha/\beta$, and $\alpha+\beta$ [2]. Structural classification of proteins (SCOP) [3] groups the proteins according to their structures and amino acid sequences and provides detailed information about the structural relationship among all recognized proteins [4]. According to SCOP, four structural classes are divided into folds. Protein fold classification determines the fold that the query protein belongs to.

[*]Correspondence: ozlem.polat@cumhuriyet.edu.tr

Proteins can be classified according to their structural classes and their folds. Classification according to the structural classes is called the first level classification, while classification according to the folds is called the second level classification. There are many studies [5–10] related to the first level classification, but in this paper we focus on the second level, namely protein fold classification. In the literature, there are different types of studies about protein fold classification. One of the oldest works dealing with this subject was conducted by Anfinsen and Scheraga [11,12]. In that work, they used the least free energy principle to solve the problem. Dubchak et al. used a neural network-based method containing three depictors related to five amino acid attributes called composition, transition, and distribution [4,13]. One of the basic studies related to fold classification was performed by Ding and Dubchak. They used support vector machine (SVM) and neural network methods to classify protein folds [14]. Ding and Dubchak prepared a protein database including 27 fold classes, and in a series of studies this dataset was used to find the fold class of the query protein [15–22]. In these studies, classification performances were calculated for each of the 27 folds. Bologna and Appel used a neural network model called discretized interpretable multi-layer perceptrons (DIMLP) [15]. Chen and Kurgan classified the protein folds by using evolutionary information and predicted secondary structure [16]. Another study was performed by Nanni. In that work Nanni proposed a new ensemble of K-local hyperplane based on random subspace and feature selection and called that method specialized ensemble (SE) [17]. To predict protein folds Okun used a classifier, which is an improved nearest neighbor method, called the k-local hyperplane distance nearest neighbor (HKNN) [18]. Shen and Chou conducted two different studies for protein fold classification. In the first one, they used optimized evidence-theoretic k-nearest neighbors (OET-KNN) for constituent individual classifiers. In the second one, they proposed a novel approach called PFP-FunDSeqE [19,20]. Kavousi et al. extracted ten different features from protein sequences and then used ten OET-KNN classifiers for the classification [21,22]. Yang et al. [23] proposed a novel margin-based ensemble classifier, called MarFold, for multiclass protein fold recognition task where multiple heterogeneous feature spaces were available. They built this method on three component classifiers, namely adaptive local hyperplane (ALH), SVM, and ALHK (a variant of ALH). Some researchers did not calculate the individual fold's success rate; they calculated only the overall success rate. For example, Suvarnavani et al. [24] applied boosting algorithm (SMOTE) to rebalance the imbalanced dataset to boost the performance and then they used a decision tree classifier to classify folds from the features of the contact map. They obtained over 70% accuracy for the feature set generated by triangle subdivision. Another study was conducted by Chmielnicki and Stapor [25]. They suggested a hybrid discriminative/generative approach. Accordingly, they combined the well-known SVM classifier with regularized discriminant analysis (RDA). In this method, SVM classified the proteins using the results of RDA, and 77.9% classification performance was achieved. In a recent study, Lin et al. [26] utilized a K-means clustering algorithm to choose a series of different base classifiers and a circulating, combined static selective strategy. Tests were performed on the dataset in [14] and 74.2% accuracy rate was obtained. Another up-to-date study was performed by Aram et al. [27] in 2015. They used a two-layer classification framework (TLCF) and a fusion of MLP, RBFN (radial basis function network), and rotation forest. In the first layer they classified the proteins according to their structural classes and in the second layer according to their folds. The fusion method was performed on the dataset in [14] and 65.7% prediction accuracy was obtained.

In this paper we propose GAL neural network [28] to classify the 27 most common SCOP folds. Compared to the other approaches in the literature, classification performance is increased by using the GAL network.

The organization of the paper is as follows: in Section 2 the dataset and features are expressed, and the

GAL network is described. The one-versus-others method and accuracy measure are explained here. In Section 3, experiments and results are shown and Section 4 concludes the paper.

## 2. Materials and methods

### 2.1. Dataset and features

The dataset used in this paper was prepared by Ding and Dubchak [14]. It is still available at http://ranger.uta.edu/$\sim$chqding/bioinfo.html. There are 313 and 385 proteins in the training and test datasets, respectively. However, two proteins in both datasets, hence a total of four proteins, do not have sequence records. For this reason, there are 311 and 383 proteins in the training and test sets. None of the proteins in the test set has more than 35% sequence similarity with the ones in the training set [14]. All these proteins belong to 27 folds included in four structural classes. Table 1 shows these folds. Structural classes of 1-6, 7-15, 16-24, and 25-27 folds are all-$\alpha$, all-$\beta$, $\alpha/\beta$, and $\alpha+\beta$, respectively. Thus, classification of proteins according to their folds is one level deeper than the classification of proteins according to their structural classes [7,29–31]. Hence, it is harder to determine the protein fold among the 27 fold classes than the protein structural class among the four structural classes [6,32].

**Table 1.** The most common 27 SCOP folds, structural classes that the folds belong to, and the number of proteins contained in training and test sets.

| Fold no. | Fold name | Str. class | Train | Test |
|---|---|---|---|---|
| 1 | Globin-like | all-$\alpha$ | 13 | 6 |
| 2 | Cytochrome c | all-$\alpha$ | 7 | 9 |
| 3 | DNA-binding 3-helical bundle | all-$\alpha$ | 12 | 20 |
| 4 | 4-helical up-and-down bundle | all-$\alpha$ | 7 | 8 |
| 5 | 4-helical cytokines | all-$\alpha$ | 9 | 9 |
| 6 | Alpha; EF-hand | all-$\alpha$ | 6 | 9 |
| 7 | Immunoglobulin-like $\beta$-sandwich | all-$\beta$ | 30 | 44 |
| 8 | Cupredoxins | all-$\beta$ | 9 | 12 |
| 9 | Viral coat and capsid proteins | all-$\beta$ | 16 | 13 |
| 10 | ConA-like lectins/glucanases | all-$\beta$ | 7 | 6 |
| 11 | SH3-like barrel | all-$\beta$ | 8 | 8 |
| 12 | OB-fold | all-$\beta$ | 13 | 19 |
| 13 | Trefoil | all-$\beta$ | 8 | 4 |
| 14 | Trypsin-like serine proteases | all-$\beta$ | 9 | 4 |
| 15 | Lipocalins | all-$\beta$ | 9 | 7 |
| 16 | (TIM)-barrel | $\alpha/\beta$ | 29 | 48 |
| 17 | FAD (also NAD)-binding motif | $\alpha/\beta$ | 11 | 12 |
| 18 | Flavodoxin-like | $\alpha/\beta$ | 11 | 13 |
| 19 | NAD(P)-binding Rossmann-fold | $\alpha/\beta$ | 13 | 27 |
| 20 | P-loop containing nucleotide | $\alpha/\beta$ | 10 | 12 |
| 21 | Thioredoxin-like | $\alpha/\beta$ | 9 | 8 |
| 22 | Ribonuclease H-like motif | $\alpha/\beta$ | 10 | 12 |
| 23 | Hydrolases | $\alpha/\beta$ | 11 | 7 |
| 24 | Periplasmic binding protein-like | $\alpha/\beta$ | 11 | 4 |
| 25 | $\beta$-grasp | $\alpha+\beta$ | 7 | 8 |
| 26 | Ferredoxin-like | $\alpha+\beta$ | 13 | 27 |
| 27 | Small inhibitors, toxins, lectins | $\alpha+\beta$ | 13 | 27 |

To deal with the protein fold classification problem, Ding and Dubchak employed physicochemical properties of amino acids. They obtained six attributes from the protein sequences. These attributes are amino acid composition, predicted secondary structure, hydrophobicity, normalized van der Waals volume, polarity, and polarizability [14]. Briefly, amino acid composition is the histogram of the 20 amino acids and shows the frequency of existence of each amino acid in a given protein. Predicted secondary structure shows the type of secondary structure, e.g., $\alpha$-helix, $\beta$-strand, or turn. Hydrophobicity is the property of repelling water rather than absorbing it or dissolving in it. Hydrophobic molecules do not like water and cluster together in it. The van der Waals volume, also called the atomic volume or molecular volume, is the atomic property most directly related to the van der Waals radius. It is the volume occupied by an individual atom (or molecule). Polarity refers to a separation of charge. In a polar molecule, electrons do not have an equal distribution in the orbital. Polarizability is the ability to constitute instant dipoles. From these six attributes, only amino acid composition contains 20 components, for the remaining five attributes each has 21 components [19]. These attributes are shown in Table 2.

**Table 2.** The six attributes that form each feature vector, their symbols, and number of components.

| Symbol | Attribute | #Components |
|---|---|---|
| C | Amino acid composition | 20 |
| S | Predicted secondary structure | 21 |
| H | Hydrophobicity | 21 |
| V | Normalized van der Waals volume | 21 |
| P | Polarity | 21 |
| Z | Polarizability | 21 |

Amino acids are generally shown with a single letter. The 20 amino acids ordered alphabetically are A,C,D,E,F, G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y. These amino acids are denoted as $AA_1$, $AA_2$,..., $AA_{20}$, and the occurrence frequency of each $AA_i$ in the given sequence is represented as $n_i$. The elements of the composition vector are obtained as follows:

$$\frac{n_1}{L}, \frac{n_2}{L}, \ldots, \frac{n_{19}}{L}, \frac{n_{20}}{L} \tag{1}$$

Here L shows the sequence length [16].

The predicted secondary structure is divided into three groups: helix, strand, and coil, and also for the other four attributes the 20 amino acids are divided into three groups depending on the size of their numerical values. For each five attributes, three depictors are calculated, namely composition (C), transition (T), and distribution (D). Composition describes the histogram of each of the three groups in a protein. Transition shows the percent frequencies related to the change between the groups. Distribution indicates the distribution of the attributes in the sequence. For each of these five attributes there are 21 scalar components. Therefore, the dimension of the feature vector for a protein is calculated as 20 + 21 × 5 = 125 [4].

### 2.2. GAL neural network

In protein fold classification, it is desired that the decision making processes lead to high performances with low computational loads, and are controlled with a few parameters. These requirements are almost satisfied by the GAL neural network [28]. For this reason, we used the GAL network to classify the 27 SCOP folds using 125 dimensional data formed by six attributes.

GAL is a growing neural network model and uses a supervised learning technique, and determines the number of nodes (neurons) during training if need arises. The network grows when it learns and shrinks when it forgets [28]. GAL represents the distribution of feature vectors according to the minimum distance measure. Computational loads of training and classification processes of GAL are rather low. Moreover, there is no parameter to be determined before the training.

### 2.2.1. GAL network's structure

The structure of the GAL neural network is portrayed in Figure 1. The first layer is composed of the input nodes. The second layer is the layer of the exemplars and the third is the class layer. The number of nodes in the exemplar layer is automatically determined during the training. When an input feature vector X is presented to the network, the distances between X and the weight vectors (W$_j$) of exemplars, $E_j$ nodes, are computed using a suitable metric, e.g., Euclidean distance. The winner-takes-all ensures that only one node will be activated, namely the node whose weight vector is closest to the input vector is determined as the winner node [28]. The network's structure is described by the following equations:

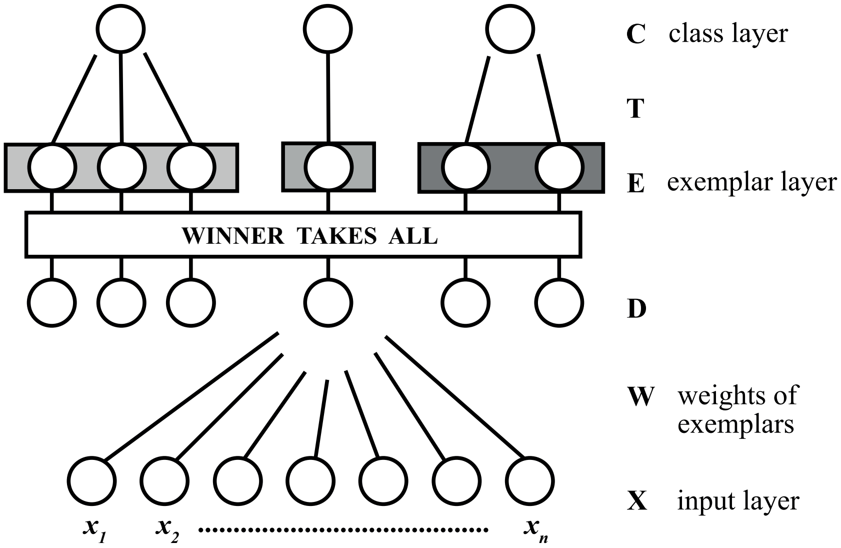

**Figure 1.** GAL network structure [28].

$$D_j = \sum_{i=1}^{n} (x_i - w_{ji})^2$$

$$E_e = \begin{cases} 1, & \text{if } D_e = \min_j D_j \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$T_{ec} = \begin{cases} 1, & \text{if } e \text{ is an exemplar of class } c \\ 0, & \text{otherwise} \end{cases}$$

$$C_c = \sum_e E_e \cdot T_{ec},$$

where $n$ is the dimension of feature space. $x_i$ denotes the $i$-th component of input feature vector X. $\mathrm{W}_j$ represents the weight vector of the exemplar node $E_j$, and $w_{ji}$ denotes the $i$-th element of $\mathrm{W}j$. $D_j$ is the distance between input vector X and $\mathrm{W}_j$. Only one of $E_j$, whose weight vector is closest to the input vector, is active (according to Eq. (2), $E_e$ is the activated exemplar). This $E_e$ activates the corresponding class node. C is the layer of class nodes. $T_{ec}$ is the connection between $E_e$ (exemplar $e$) to class $c$. $T_{ec}$ values will be 1 or 0 depending on whether $E_e$ is an exemplar of class $c$ or not. These connections are initially set to zero, and become 1 during the training. The activations of class nodes are computed by a dot product. Depending on the distribution of the training vectors in the feature space, more than one exemplar node can be associated with each class node (note that the exemplars of each class are grouped within boxes in Figure 1). Therefore, any exemplar of a class can activate its corresponding class node.

### 2.2.2. Partitioning of feature space by the GAL network

As an example to depict how GAL partitions the feature space, a two-dimensional phantom feature space is formed; see Figure 2. In this phantom space, it is seen that there are two different classes each having three nodes (exemplars). As the nearest distance measure is used in the classification, a hyperplane passes through the two closest nodes that belong to different classes, and this hyperplane is equidistant to both nodes (while we call it a *hyperplane* in an $n$-dimensional space, it is a *line* in two dimensions). A piecewise class boundary is thus created with several hyperplanes. The same mechanism is valid in multidimensional feature spaces.

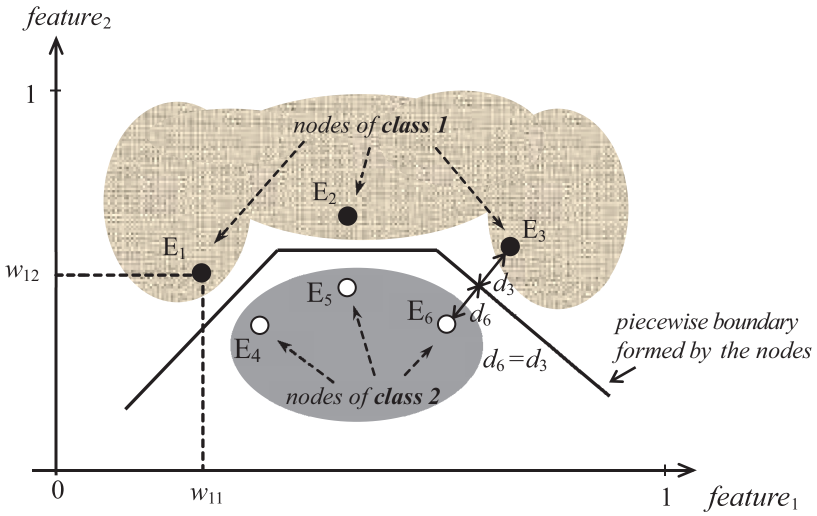

**Figure 2.** GAL's partitioning of a two-dimensional phantom feature space.

### 2.3. Learning and forgetting in GAL

The most important feature of GAL is that the number of $E_j$ nodes is automatically determined and gradually increased during learning. The choice of exemplar layer nodes, and hence the structure of the GAL network, differs according to the order of the initially given input vectors. A node stored in previous iterations may become redundant when a new node closer to the class boundary is produced. In order to keep the topology of the network simple, those redundant nodes may be excluded from the network with the forgetting algorithm of GAL. The aim of the forgetting algorithm is to detect and exclude those nodes that do not change the performance of the network when being eliminated. The steps during the learning are mentioned below [28,33].

*Step 1*: Initially select a number of feature vectors randomly from the training set as many as the number of classes. Each vector should be selected among the patterns of a particular class. Set each chosen vector as an exemplar layer node of GAL. Set the iteration number.

*Step 2*: Decrease the iteration number. If the iteration number is equal to zero terminate the learning algorithm, otherwise go to *Step 3*.

*Step 3*: Take a feature vector randomly from the training set, and present it to the network as input.

*Step 4*: Calculate the distances between the input vector and the exemplar layer nodes and determine the closest node according to Eq. (2). If the classes of the closest node and input vector are the same, go to *Step 2*. Otherwise, go to *Step 5*.

*Step 5*: Include the input vector as a new exemplar node to the network. The input vector is assigned as the associated weight vector of the new node. Go to *Step 2*.

The forgetting algorithm can be run several times during the training depending on the iteration number. The steps during the forgetting are given below. Iteration number is initialized as the number of exemplar layer nodes.

*Step 1*: Temporarily remove an exemplar node from the network in some order and present this node as an input vector to the network.

*Step 2*: Calculate the distances between input vector and exemplar layer nodes. If the classes of the input vector and the closest node are the same, go to *Step 4*.

*Step 3*: Put the input vector back in the network.

*Step 4*: Decrease the iteration number. If the iteration number is equal to zero terminate the forgetting algorithm, otherwise go to *Step 1*.

### 2.4. One-versus-others (OvO) method and performance measures

The OvO prediction method is an easily applicable and efficient method [4,34], and it is generally used in multiclass problems. To explain this method, suppose that there are $K$ classes. Firstly, we transform the multiclass problem into a two-class problem. One class contains all the proteins belonging to the $i$-th fold that are labeled as *positive*, and the other class contains all other proteins that are labeled as *negative*. Thus we construct $K$ binary classifiers to predict the protein folds. For example, in the first classifier one class contains the first fold's proteins and the other class contains the other $K-1$ folds' proteins.

In the recognition process, the new query protein is tested at each of the $K$ binary classifiers. Thus, $K$ scores are obtained from the $K$ classifiers. Ideally, it is expected that only one of the $K$ classifiers gives a positive result and the others give negative results, so that the query protein is assigned to a unique fold [14].

In this study, as we try to solve the 27-class protein fold classification problem, the number of binary classifiers is 27 ($K = 27$). We performed various tests to quantify the performance of the classifier. To calculate the classification success rates we used sensitivity (true positive rate, TPR) and accuracy. During the tests using OvO, we used sensitivity (Eq. (3)) to calculate an individual fold's success rate and we used accuracy (Eq. (4)) to calculate the overall success rate.

$$\text{Ind. Fold's Success Rate} = \frac{\sum \text{True Positive}}{\sum \text{True Positive} + \sum \text{False Negative}} \tag{3}$$

$$\text{Overall Success Rate} = \frac{\sum_{i=1}^{27} \text{True Positive}}{\sum_{i=1}^{27} \text{True Positive} + \sum_{i=1}^{27} \text{False Negative}} \tag{4}$$

In some tests we did not use the OvO method and we calculated the classifier's success rate using the accuracy measure below:

$$\text{Accuracy} = \frac{\sum \text{True Positive} + \sum \text{True Negative}}{\sum \text{Positive} + \sum \text{Negative}} \tag{5}$$

## 3. Results

In the first experiment, we tested the proposed fold classification method and we calculated the success rate using Eq. (5). Then, in order to increase the classification performance, we used the OvO method by implementing 27 binary classifiers and calculated the success rate using Eq. (4). After these tests, we applied 10-fold cross-validation (CV) to get unbiased training data for the classification of proteins, and, as before, we tested the proteins with and without OvO. For 10-fold CV, the training set is partitioned into ten folds with each fold containing almost equal number of patterns of each class. As expected, by incorporating the 10-fold CV, the success rates increased for both tests (tests with and without OvO). The comparisons of the results are shown in Table 4. According to the test results, it is seen that, when we used just one classifier to classify the 27 folds, we obtained a poor success rate of 44.1%. After applying 10-fold cross-validation, the success rate increased from 44.1% to 57.1%, which roughly corresponds to a 30% performance increment. Then, to further increase the performance, the OvO method is employed. With the use of OvO, 27 binary classifiers are formed to classify the 27 folds, and the overall success rate is obtained as 81.2%. Again, in order to test the GAL classifier on an unbiased dataset, we used 10-fold cross validation with the OvO method, and observed the success rate to be increased to 87.7%.

**Table 3.** Five attributes and the division into three groups. While the first attribute is related to the secondary structures, the other four attributes are related to amino acids.

| Property | Group 1 | Group 2 | Group 3 |
|---|---|---|---|
| S | Helix | Strand | Coil |
| H | Polar<br>R,K,E,D,Q,N | Neutral<br>G,A,S,T,P,H,Y | Hydrophobic<br>C,V,L,I,M,F,W |
| V | 0-2.78<br>G,A,S,C,T,P,D | 2.95-4.0<br>N,V,E,Q,I,L | 4.43-8.08<br>M,H,K,F,R,Y,W |
| P | 4.9-6.2<br>L,I,F,W,C,M,V,Y | 8.0-9.2<br>P,A,T,G,S | 10.4-13.0<br>H,Q,R,K,N,E,D |
| Z | 0-0.108<br>G,A,S,D,T | 0.128-0.186<br>C,P,N,V,E,Q,I,L | 0.219-0.409<br>K,M,H,F,R,Y,W |

**Table 4.** Prediction methods used with GAL and the related success rates.

| Test methods | Success rate (%) |
|---|---|
| Accuracy | 44.1 |
| 10-fold CV + Accuracy | 57.1 |
| OvO + Sensitivity | 81.2 |
| 10-fold CV + OvO +Sensitivity | 87.7 |

**Table 5.** True positive rates for individual folds and overall success rates using GAL coupled with the OvO method for a few combinations of the six attributes.

| Fold no. | C | CS | CSH | CSHV | CSHVP | CSHVPZ |
|---|---|---|---|---|---|---|
| 1 | 83.3 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| 2 | 77.8 | 88.9 | 88.9 | 100.0 | 100.0 | 100.0 |
| 3 | 60.0 | 75.0 | 75.0 | 85.0 | 85.0 | 80.0 |
| 4 | 87.5 | 87.5 | 100.0 | 100.0 | 100.0 | 87.5 |
| 5 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| 6 | 66.7 | 66.7 | 66.7 | 55.6 | 55.6 | 77.8 |
| 7 | 65.9 | 65.9 | 72.7 | 68.2 | 70.5 | 70.5 |
| 8 | 50.0 | 66.7 | 66.7 | 75.0 | 75.0 | 75.0 |
| 9 | 92.3 | 76.9 | 76.9 | 76.9 | 84.6 | 84.6 |
| 10 | 66.7 | 66.7 | 83.3 | 83.3 | 83.3 | 66.7 |
| 11 | 62.5 | 75.0 | 87.5 | 100.0 | 87.5 | 87.5 |
| 12 | 36.8 | 52.6 | 47.4 | 47.4 | 47.4 | 52.6 |
| 13 | 75.0 | 75.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| 14 | 75.0 | 100.0 | 100.0 | 100.0 | 100.0 | 75.0 |
| 15 | 85.7 | 71.4 | 85.7 | 85.7 | 71.4 | 85.7 |
| 16 | 79.2 | 83.3 | 83.3 | 83.3 | 87.5 | 85.4 |
| 17 | 66.7 | 83.3 | 83.3 | 91.7 | 83.3 | 75.0 |
| 18 | 61.5 | 69.2 | 69.2 | 76.9 | 69.2 | 84.6 |
| 19 | 66.7 | 59.3 | 59.3 | 66.7 | 51.9 | 81.5 |
| 20 | 83.3 | 75.0 | 75.0 | 66.7 | 75.0 | 75.0 |
| 21 | 62.5 | 75.0 | 87.5 | 87.5 | 87.5 | 87.5 |
| 22 | 91.7 | 91.7 | 91.7 | 83.3 | 91.7 | 91.7 |
| 23 | 85.7 | 85.7 | 100.0 | 100.0 | 100.0 | 100.0 |
| 24 | 75.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |
| 25 | 50.0 | 75.0 | 75.0 | 75.0 | 87.5 | 75.0 |
| 26 | 51.9 | 59.3 | 59.3 | 55.6 | 70.4 | 66.7 |
| 27 | 100.0 | 92.6 | 92.6 | 88.9 | 96.3 | 100.0 |
| Overall | 71.3 | 75.2 | 77.5 | 78.1 | 79.4 | 81.2 |

C refers to amino acid composition
S refers to predicted secondary structure
H refers to hydrophobicity
V refers to van der Waals volume
P refers to polarity
Z refers to polarizability

To determine the effectiveness of the features we performed some tests as in [14]. Firstly, we used only the C attribute to be contained in the feature vectors. Then we appended the S attribute to C and so we used C+S as the components of the feature vectors; progressively in the last test we used all the six attributes to form the feature vectors. In all these tests, iteration number was set to 3000 during the training of the GAL classifier. The results related to this set of tests are shown in Table 5. The overall success rate increases very substantially, from 71.3% for the amino acid composition attribute to 75.2% for amino acid composition + predicted secondary structure attributes. It increases from 75.2% to 77.5% for amino acid composition + predicted secondary structure + hydrophobicity attributes. By continuing in this way classification performance reaches 81.2% for all six attributes. According to this table we observed that the amino acid composition (C),

**Table 6.** Classification rates for individual folds and the overall success rates for different methods.

| Fold | [14] | [18] | [15] | [17] | [19] | [21] | [16] | [20] | [23] | [22] | GAL |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 83.3 | 83.3 | 85.0 | 83.3 | 83.3 | 100 | 100 | 100 | 83.3 | 100 | 100 |
| 2 | 77.8 | 77.8 | 97.8 | 88.9 | 55.6 | 100 | 100 | 88.9 | 100 | 100 | 100 |
| 3 | 35.0 | 50.0 | 66.0 | 70.0 | 85.0 | 75.0 | 60.0 | 60.0 | 70.0 | 90.0 | 80.0 |
| 4 | 50.0 | 87.5 | 41.3 | 50.0 | 75.0 | 62.5 | 75.0 | 87.5 | 87.5 | 100 | 87.5 |
| 5 | 100 | 88.9 | 91.1 | 100 | 100 | 100 | 88.9 | 77.8 | 100 | 77.8 | 100 |
| 6 | 66.7 | 44.4 | 22.2 | 33.3 | 33.3 | 55.6 | 66.7 | 66.7 | 55.6 | 77.8 | 77.8 |
| 7 | 71.6 | 56.8 | 75.7 | 79.6 | 70.5 | 77.3 | 81.8 | 77.3 | 95.5 | 63.6 | 70.5 |
| 8 | 16.7 | 25.0 | 40.0 | 25.0 | 16.7 | 25.0 | 33.3 | 75.0 | 25.0 | 75.0 | 75.0 |
| 9 | 50.0 | 84.6 | 80.8 | 69.2 | 100 | 69.2 | 92.3 | 92.3 | 76.9 | 76.9 | 84.6 |
| 10 | 33.3 | 50.0 | 46.7 | 33.3 | 33.3 | 66.7 | 66.7 | 66.7 | 50.0 | 33.3 | 66.7 |
| 11 | 50.0 | 50.0 | 75.0 | 62.5 | 37.5 | 37.5 | 62.5 | 37.5 | 75.0 | 87.5 | 87.5 |
| 12 | 26.3 | 42.1 | 22.6 | 36.8 | 15.8 | 26.3 | 52.6 | 42.1 | 36.8 | 78.9 | 52.6 |
| 13 | 50.0 | 50.0 | 45.0 | 50.0 | 75.0 | 100 | 75.0 | 100 | 75.0 | 100 | 100 |
| 14 | 25.0 | 50.0 | 50.0 | 25.0 | 50.0 | 25.0 | 50.0 | 75.0 | 50.0 | 75.0 | 75.0 |
| 15 | 57.1 | 42.9 | 74.3 | 28.6 | 71.4 | 100 | 100 | 100 | 71.4 | 85.7 | 85.7 |
| 16 | 77.1 | 79.2 | 83.8 | 87.5 | 97.9 | 91.7 | 68.8 | 72.9 | 87.5 | 25.0 | 85.4 |
| 17 | 58.3 | 58.3 | 55.0 | 58.3 | 66.7 | 75.0 | 91.7 | 91.7 | 83.3 | 75.0 | 75.0 |
| 18 | 48.7 | 53.9 | 52.3 | 61.5 | 15.4 | 15.4 | 46.2 | 61.5 | 61.5 | 69.2 | 84.6 |
| 19 | 61.1 | 40.7 | 39.3 | 37.0 | 44.4 | 70.4 | 66.7 | 66.7 | 55.6 | 81.5 | 81.5 |
| 20 | 36.1 | 33.3 | 41.7 | 50.0 | 33.3 | 50.0 | 33.3 | 50.0 | 50.0 | 66.7 | 75.0 |
| 21 | 50.0 | 37.5 | 46.3 | 50.0 | 62.5 | 87.5 | 50.0 | 87.5 | 75.0 | 100 | 87.5 |
| 22 | 35.7 | 71.4 | 55.0 | 64.3 | 66.7 | 50.0 | 66.7 | 75.0 | 64.3 | 75.0 | 91.7 |
| 23 | 71.4 | 71.4 | 44.3 | 71.4 | 57.1 | 57.1 | 57.1 | 71.4 | 71.4 | 85.7 | 100 |
| 24 | 25.0 | 25.0 | 25.0 | 25.0 | 50.0 | 25.0 | 50.0 | 100 | 25.0 | 50.0 | 100 |
| 25 | 12.5 | 25.0 | 23.8 | 25.0 | 37.5 | 37.5 | 25.0 | 25.0 | 25.0 | 75.0 | 75.0 |
| 26 | 37.0 | 25.9 | 41.1 | 33.3 | 29.6 | 11.1 | 51.9 | 33.3 | 55.6 | 85.2 | 66.7 |
| 27 | 83.3 | 85.2 | 100 | 85.2 | 96.3 | 100 | 96.3 | 96.3 | 100 | 100 | 100 |
| Overall | 56.0 | 57.1 | 61.1 | 61.1 | 62.1 | 67.2 | 68.4 | 70.5 | 71.7 | 73.1 | 81.2 |

[14] refers to Ding and Dubchak (2001)
[15] refers to Bologna and Appel (2002)
[16] refers to Chen and Kurgan (2007)
[17] refers to Nanni (2006)
[18] refers to Okun (2004)
[19] refers to Shen and Chou (2006)
[20] refers to Shen and Chou (2009)
[21] refers to Kavousi et al. (2011)
[22] refers to Kavousi et al. (2012)
[23] refers to Yang et al. (2011)

even tested alone, gives a reasonable success rate (71.3%), but it is obvious that the highest success rate can yet be achieved by the contribution of all six attributes.

In order to exhibit the competency of GAL, tests were performed on the same protein dataset used by many researchers [14–22]. The overall success rate and individual folds' success rates related to the 27-class fold classification task are given in Table 6. This table shows the classification results of the GAL network using OvO and compares GAL with the previous studies such as support vector machine [14], discretized interpretable multi-layer perceptrons [15], PFRES [16], specialized ensemble [17], hyperplane distance nearest neighbor algorithm [18], PFP-Pred [19], PFP-FunDSeqE [20], information theoretic classifier fusion [21], and hyperfolds [22]. Each row in this table indicates the success rate of an individual fold and the last row indicates the overall success rate. To calculate individual folds' success rates and overall success rate, Eqs. (3) and (4) are used respectively. While applying GAL, iteration number was determined as 3000. At the end of the individual folds' training, an average of 31 nodes were generated to classify each fold. According to Table 6, the 1st, 2nd, 5th, 13th, 23rd, 24th, and 27th folds are classified with 100% accuracy using the GAL network. As seen from the table, the overall success rate of the GAL network is remarkably higher than those of the other existing methods and it is the only classifier having a success rate above 80%. Moreover, the success rate of the GAL network for each fold does not fall below 52.6%, while the success rates of all the other methods in Table 6 are far below this value for at least two folds.

## 4. Conclusions

In this paper we consider protein fold classification and propose a new solution to the 27-class protein fold classification problem using GAL network. GAL basically is a variant of the nearest neighbor method. The network is modified during training. Successive learning and forgetting phases allow the system to choose a good subset for classification. In this study we also used the popular OvO method. This method improved the classification success rate. Overall, we obtained an 81.2% success rate for the protein fold classification problem.

GAL's learning algorithm tends to generate slightly more nodes when the boundaries of classes get closer. In fact, closeness of patterns from different classes is an indication that the chosen features are not successful enough in representing the within-class and between-class properties of classes in the feature space. Therefore, it is more conceivable to search for new feature extraction methods. However, in this study, after the training is completed, it is observed that for 27 protein folds the GAL network trained with 694 protein patterns has generated an average of 31 nodes for each of the 27 binary classifiers. The number of generated nodes can be regarded as reasonable with such a big 27-class problem. Hence, the 125 features composed from six attributes seem promising for the protein fold classification problem.

In addition, we have studied the effectiveness of the six attributes. We have tested them by incorporating the six attributes gradually, with the order 'C', 'S', 'H', 'V', 'P', and 'Z', and the results showed that the amino acid composition (C) attribute (having 20 components), even tested alone, gives a reasonable success rate of 71.3%. When we consider the dimension of the feature vectors, achieving a success of 71.3% with only 20 components is comparable with the existing methods in the literature as they can reach this level of success rates with much more features. However, it is obvious that to obtain higher rates with GAL, we still need to incorporate all 125 features composed from six attributes.

In this study we aimed to distinguish different protein folds using GAL network. In future works, this neural network model will be tested for classification of more than 27 folds with larger datasets. It will be more challenging to deal with larger datasets in such huge dimensional feature spaces. Thus, we will focus on reducing the dimension of current feature vectors or searching for new feature extraction methods.

## References

[1] Hashemi HB, Shakery A, Naeini MP. Protein fold pattern recognition using Bayesian ensemble of RBF neural networks. In: IEEE 2009 Soft Computing and Pattern Recognition Conference; 4–7 December 2009; Malacca, Maleysia. IEEE. pp. 436-441.

[2] Levitt M, Chothia C. Structural patterns in globular proteins. Nature 1976; 27: 254-256.

[3] Murzin AG, Brenner SE, Hubbard T, Chothia C. SCOP: a structural classifications of proteins database for the investigation of sequences and structures. J Mol Biol 1995; 247: 536-540.

[4] Dubchak I, Muchnik I, Mayor C, Dralyuk I, Kim SH. Recognition of a protein fold in the context of the SCOP classification. Proteins 1999; 35: 401-407.

[5] Cai YD, Zhou GP. Prediction of protein structural classes by neural network. Biochimie 2000; 82: 783-785.

[6] Chou KC. A novel approach to predicting protein structural classes in a (20-1)-D amino acid composition space. Proteins 1995; 21: 319-344.

[7] Chou KC, Zhang CT. Review: prediction of protein structural classes. Crit Rev Biochem Mol Biol 1995; 30: 275-349.

[8] Isik Z, Yanikoglu B, Sezerman U. Protein structural class determination using support vector machines. Lecture Notes in Computer Science 2004; 3280: 82-89.

[9] Sun XD, Huang RB. Prediction of protein structural classes using support vector machine. Amino Acids 2006; 30: 469-475.

[10] Zhang CT, Chou KC. An optimization approach to predicting protein structural class from amino acid composition. Protein Sci 1992; 1: 401-408.

[11] Anfinsen CB. Principles that govern the folding of protein chains. Science 1973; 181: 223-230.

[12] Anfinsen CB, Scheraga HA. Experimental and theoretical aspects of protein folding. Adv Protein Chem 1975; 29: 205-300.

[13] Dubchak I, Muchnik I, Holbrook SN, Kim SH. Prediction of protein folding class using global description of amino acid sequence. P Natl Acad Sci USA 1995; 92: 8700-8704.

[14] Ding CHQ, Dubchak I. Multi-class protein fold recognition using support vector machines and neural networks. Bioinformatics 2001; 17: 349-358.

[15] Bologna G, Appel RD. A comparison study on protein fold recognition. In: IEEE 2002 9th International Conference on Neural Information Processing; 18–22 Nov 2002; IEEE. pp. 2492-2496.

[16] Chen K, Kurgan L. PFRES: protein fold classification by using evolutionary information and predicted secondary structure. Bioinformatics 2007; 23: 2843-2850.

[17] Nanni L. A novel ensemble of classifiers for protein fold recognition. Neurocomputing 2006; 69: 2434-2437.

[18] Okun O. Protein fold recognition with k-local hyperplane distance nearest neighbour algorithm. In: Proceedings of the Second European Workshop on Data Mining and Text Mining in Bioinformatics 2004; Pisa, Italy. Citeseer. pp. 51-57.

[19] Shen HB, Chou KC. Ensemble classifier for protein fold pattern recognition. Bioinformatics 2006; 22: 1717-1722.

[20] Shen HB, Chou KC. Predicting protein fold pattern with functional domain and sequential evolution information. J Theor Biol 2009; 256: 441-446.

[21] Kavousi K, Moshiri B, Sadeghi M, Araabi BN, Moosavi-Movahedi AA. A protein fold classifier formed by fusing different modes of pseudo amino acid composition via PSSM. Comput Biol Chem 2011; 35: 1-9.

[22] Kavousi K, Sadeghi M, Moshiri B, Araabi BN, Moosavi-Movahedi AA. Evidence theoretic protein fold classification based on the concept of hyperfold. Math Biosci 2012; 240: 148-160.

[23] Yang T, Kecman V, Cao L, Zhang C, Huang JZ. Margin-based ensemble classifier for protein fold recognition. Expert Syst Appl 2011; 38: 12348-12355.

[24] Suvarnavani K, Rafiah SB, Kamisetti NR. Multiclass classification for protein fold prediction using Smote. Int J Adv Res Comput Sci Softw Eng 2012; 2: 290-296.

[25] Chmielnicki W, Stapor K. A hybrid discriminative/generative approach to protein fold recognition. Neurocomputing 2012; 75: 194-198.

[26] Lin C, Zou Y, Qin J, Liu X, Jiang Y, Ke C, Zou Q. Hierarchical classification of protein folds using a novel ensemble classifier. PloS ONE 2013; 8: 1-11.

[27] Aram RZ, Charkari NM. Two-layer classification framework for protein fold recognition. J Theor Biol 2015; 365: 32-39.

[28] Alpaydin E. GAL: Networks that grow when they learn and shrink when they forget. Int J Pattern Recogn 1991; 8: 391-414.

[29] Cai YD. Is it a paradox or misinterpretation. Proteins 2001; 43: 336-338.

[30] Zhou GP. An intriguing controversy over protein structural class prediction. J Protein Chem 1998; 17: 729-738.

[31] Zhou GP, Assa-Munt N. Some insights into protein structural class prediction. Proteins 2001; 44: 57-59.

[32] Chou KC, Maggiora GM. Domain structural class prediction. Protein Eng 1998; 11: 523-538.

[33] Dokur Z, Ölmez T. Classification of respiratory sounds by using an artificial neural network. Int J Pattern Recogn 2003; 17: 567-580.

[34] Brown MPS, Grundy WN, Lin D, Cristianini N, Sugnet CW, Furey TS, Ares M, Haussler D. Knowledge-based analysis of microarray gene expression data by using Support Vector Machines. P Natl Acad Sci USA 2000; 97: 262-267.